

Stand for Remote Development of a Display Visualization System for Civil Aircraft Cockpit

S.V. Andreev¹, E.Yu. Denisov², I.G. Ryzhova³, A.G. Voloboy⁴

Keldysh Institute of Applied Mathematics RAS

¹ ORCID: 0000-0001-8029-1124, andreev@keldysh.ru

² ORCID: 0000-0002-0614-9100, eed@spp.keldysh.ru

³ ORCID: 0000-0003-1613-3038, ryzhova@gin.keldysh.ru

⁴ ORCID: 0000-0003-1252-8294, voloboy@gin.keldysh.ru

Abstract

The paper presents the original development of the stand for remote creation and debugging of the visualization system for the display of a civil aircraft cockpit. Any visualization system implies the presence of a person for whom it is ultimately intended. Naturally, such presence is necessary for the process of creating and testing the system. However, in the context of the COVID-19 pandemic, access to the premises of scientific institutions is limited. The situation is further complicated by the fact that the development of on-board software must take into account the specifics and be carried out on the target on-board equipment. Therefore, we designed and built a stand which provides remote secure access to the target platform, its remote control and makes possible to see the results of its work. We managed to avoid costly solutions and use a regular desktop PC as an auxiliary control computer. As a result the process of developing a visualization system for the cockpit display is not interrupted even when access to the target platform equipment is restricted.

Keywords: visualization system, pilot display, remote access, distributed software development.

1. Introduction

Any visualization system implies the presence of a person for whom it is ultimately intended. Naturally, such presence is necessary for the process of development and testing of the visualization system. However, as it has been shown in 2020, this turned out to be not always feasible.

In the critical situations of the worldwide pandemic of the COVID-19 virus, access to offices in R&D centers has been significantly limited. Thus, the issue of organization of remote work of researchers has become acute. In this situation a natural need is remote and secure access via the Internet to the local network of the institute, to its servers and workstations, for example, using VPN (Virtual Private Network). But often there is a need for access and remote control of instruments and objects of research, as well as visual observation of their behavior in the absence of direct access of researchers.

The work [1] describes the development of a visualization system for displays of the cockpit of a civil aircraft. In the modern cockpit indicators are almost completely replaced by graphic displays (Fig. 1). However, the peculiarities of this visualization system are the use of special equipment with low power consumption and work under a Real-Time OS (RTOS). The development of the system must ultimately take place on the target platform.



Fig.1. The Boeing 737-800 cockpit (Courtesy of [Cory W. Watts](#)).

Therefore, in the situations of quarantine and restricted access of the development group to the equipment, there was an urgent need to organize remote secure access to the target platform, remote control of it and the ability to see the results of its work. This required the creation of a special installation (stand) consisting of a target platform, a computer for control and operating of the stand and a system for capturing and transmitting images of the pilot's display. An important aspect of this task was the requirement to create this stand as quickly as possible and with minimal additional financing. It was therefore assumed to use, if possible, publicly available software products, i.e. it is desirable to avoid creating own specialized software. At the same time, the acceptable level of security for remote access and control of the stand via the Internet had to be provided. We also considered the possibility of using this solution in the future for other developments using that use the remote working option.

2. Related works

The task described above is quite actual. A number of works are devoted to providing remote access and control of target equipment in which various solutions are proposed for their specific conditions.

The MIT iLab project developed a distributed a service infrastructure and software toolkit to support a scalable community of online laboratory experiments [2]. The overall architecture of iLab provides a framework for developing and deploying remote labs using a three-level model based on web services consisting of lab clients, service broker auxiliary software and lab servers. Some of the further works on the creation of remote learning laboratories are based on this development.

The paper [3] describes the experience of creating an online laboratory system with the priority of modular and generalized solutions. The developed system provides a unified user

interface for accessing, managing and controlling a remote experiment. A practical case of remote laboratories sharing is described in [4]. The required functionality for user management, access, experimentation and resource sharing is provided by the developed Remote Lab Management System (RLMS) WebLab-Deusto. The paper [5] describes the creation and implementation of commercial products, tools and services for online laboratory development. It also describes real-world experience with online labs from both a technological and educational point of view. An attempt to classify existing online laboratories serving for practical training of the technical specialists was made in [6].

Two online laboratories for scientific and educational purposes were created at the Bauman Moscow State Technical University. A laser laboratory for remote access [7] is a software and hardware complex based on the technology of virtual devices which allows combination of visualization of a real physical experiment with the convenience and safety of remote control of the research course. The Internet laboratory "Robotics" [8] developed in the Dmitrov branch of the Bauman University allows you to practice control of the robots of the International Space Station. It provides the ability to conduct complex remote laboratory workshops on real complex robotic equipment using network access technologies.

The Higher School of Economics (University) has developed and created a system for remote access to the equipment of the Educational Laboratory for Computer Aided Design (CAD) Systems [9]. The original specialized software allows remote control of target boards.

The remote access stands for development and research of adaptive and intelligent control systems for suppressing elastic vibrations were created at the St. Petersburg State Electrotechnical University "LETI" named after V.I. Uljanov (Lenin) [10].

Existing solutions described here do not meet our needs for three main reasons. Firstly, they serve highly specialized purposes, i.e. they are unique, customized specifically for the solution of this specific task. Most of the systems are for educational purposes only. Secondly, they often involve the use of additional rather expensive software and / or hardware. Third, some systems do not provide access to target equipment but use virtual instruments and sensors. It is also clear that there is no overall systematic approach for providing remote access and control of target equipment.

3. The main components of the stand for remote development of the visualization system

As it was described in details in [1], the development of onboard software should take into account the specifics of the equipment. Aircraft onboard systems use reliable processors with relatively low power consumption and performance which are also resistant to temperature extremes. We worked with the NXP i.MX 6 SoC family of processors [11] when developing the stand. But our experience can be applied to other hardware. We call it "target platform" in the paper.

An important feature of the target platforms of the stand is the need to load test examples into them, running under control of the developed specialized RTOS of the on-board computer [12]. For obvious reasons the RTOS does not support any remote access and does not allow control over the Internet.

At the same time, in addition to loading such a specialized RTOS on the target platform, it becomes necessary to load the Linux operating system because its environment that is quite familiar and convenient for development and research. Developers need to be able to directly debug a software product on target platform taking into account the hardware specifics. The use of a compiler, debugger, profiler and other programming tools that are common in the Linux environment greatly simplifies development.

When the developer has direct physical access to the target platform such a task looks quite trivial. The user just needs to set the appropriate boot source for a particular OS (for example, insert a CCD memory card containing the OS) and turn on the power button of the

target device. However, the task becomes much more complicated in conditions of remote access, without physical access to equipment.

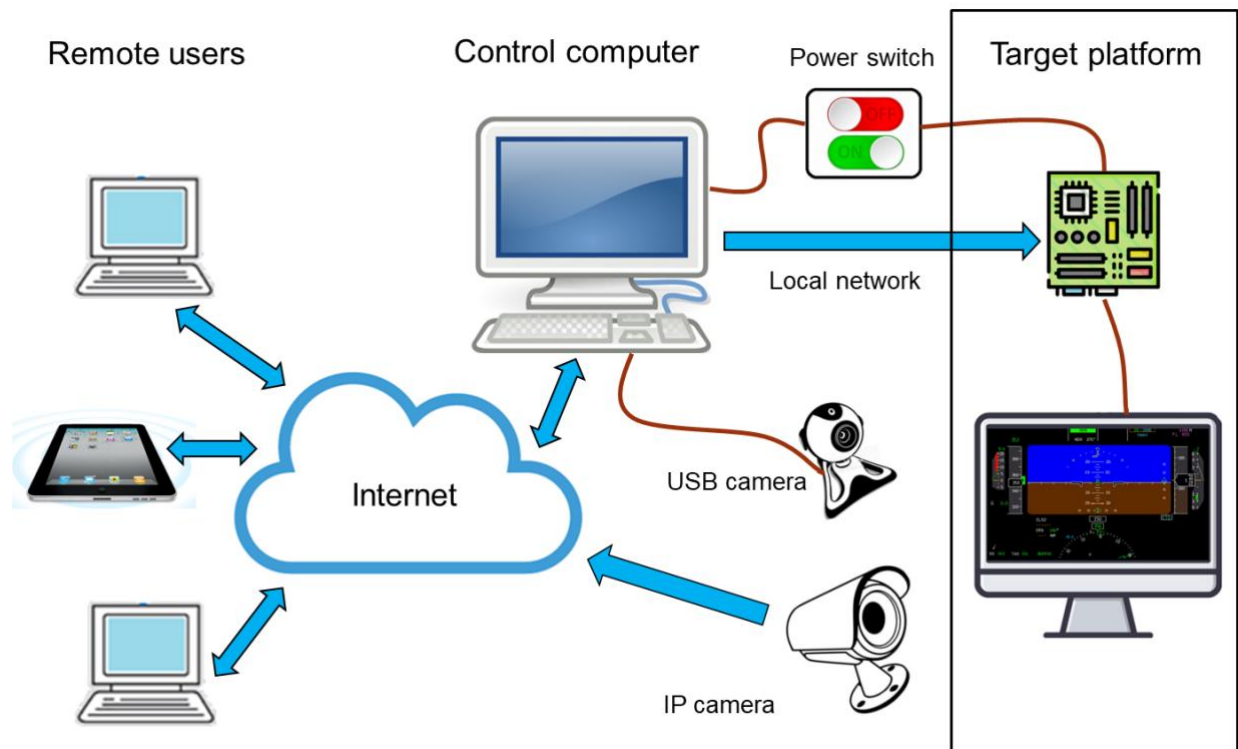


Fig. 2. The general scheme of the developed stand.

The development of a visualization system implies monitoring and observing the result on a display screen physically. It is not enough to parse the instruction set or analyze the image being sent to the GPU. Therefore, it is also necessary to provide visual observation of the results of graphic output on the display screen as well as controlling the picture using the keyboard or mouse.

All this leads to the need to use additional equipment. The auxiliary control computer provides developers with remote permanent access to the stand from home via the Internet and control of the target platform equipment, up to power off. Cameras placed in front of the monitor and aimed at it provide feedback. The general scheme of the developed stand is shown in Fig. 2.

4. The auxiliary control computer

What general criteria and requirements should a remote control workstation meet, i.e. what are the requirements to a control computer and its additional equipment, to software?

(1) Additional equipment should not be expensive and / or difficult to purchase (highly specialized). It allows to minimize additional expenses in the face of unexpected restrictions due to the COVID-19 pandemic.

(2) The control computer should provide uninterrupted remote client access to the developers' control of the target platform hardware over the Internet.

(3) Providing developers with access to the control of the target platform using a control computer (clients) should not lead to the need to purchase additional highly specialized hardware or software for all participants in the development. Developers should have possibility to connect from ordinary home computers.

(4) The client part of the access software must provide access to the control computer from different operating systems (MS Windows, Linux, MacOS) depending on what is already installed and used on the client's home computer or laptop. This will allow

developers to use their computers in their familiar environment (versatility) including using home computers for their own purposes.

(5) The access of clients (developers) to the control computer via the Internet must ensure a sufficiently high level of security, i.e. it must use an encrypted channel that cannot be intercepted by third parties.

(6) Simultaneous access of multiple clients (developers) to the control computer for collaboration should be provided (distributed development).

4.1. Interaction between control computer and target platform

Typically, the chipset of the target on-board computer is not some highly specialized and expensive equipment made exclusively for aircraft. In most cases manufacturers produce a similar chipset for quite household applications at an affordable price. For example, they are intended for home use as an economical and temperature-resistant video surveillance system, for creating a smart home control system for home equipment such as a mini web server, home theater, media server or household router. The use of such systems in everyday life is due to the extremely low power consumption of the chipset as well as high resistance to external influences and low cost.

Thus, developers of on-board computer systems can initially gain access to the target platform even when the prototype of the on-board system has not yet been created, but is at the stage of exploring its possibilities and applicability in aviation.



Fig. 3. The target platform in the form of a household microcomputer based on NXP i.MX 6 chipsets.

Typically, such a chipset comes in the form of a ready-made microcomputer with the ability to connect to a network, an external monitor, USB keyboard and mouse (Fig. 3). At the same time, manufacturers provide such a microcomputer with support for memory cards with the ability to boot the OS from a card. However, in most cases this OS is a relatively outdated version of Linux, with an old kernel and with proprietary chip manufacturer drivers (no open source). It is also often not possible to install additional packages required by developers, such as a compiler, debugger, or profiler.

Therefore, we had to create a scheme for booting the target platform OS over the network using the TFTP (Trivial File Transfer Protocol) from a console application on a control computer with the option of choosing our specially created version of the Linux OS. The control computer and the target platform must be connected to the same local network. The target platform communicates with the control computer console through a serial connection. Thus, it is possible to select from the console application of the control computer the OS to be loaded without the need to physically replace the memory card in the target platform. This is necessary for organizing remote access. The U-Boot [13] software product was used as the OS loader on the memory card (Fig. 4).

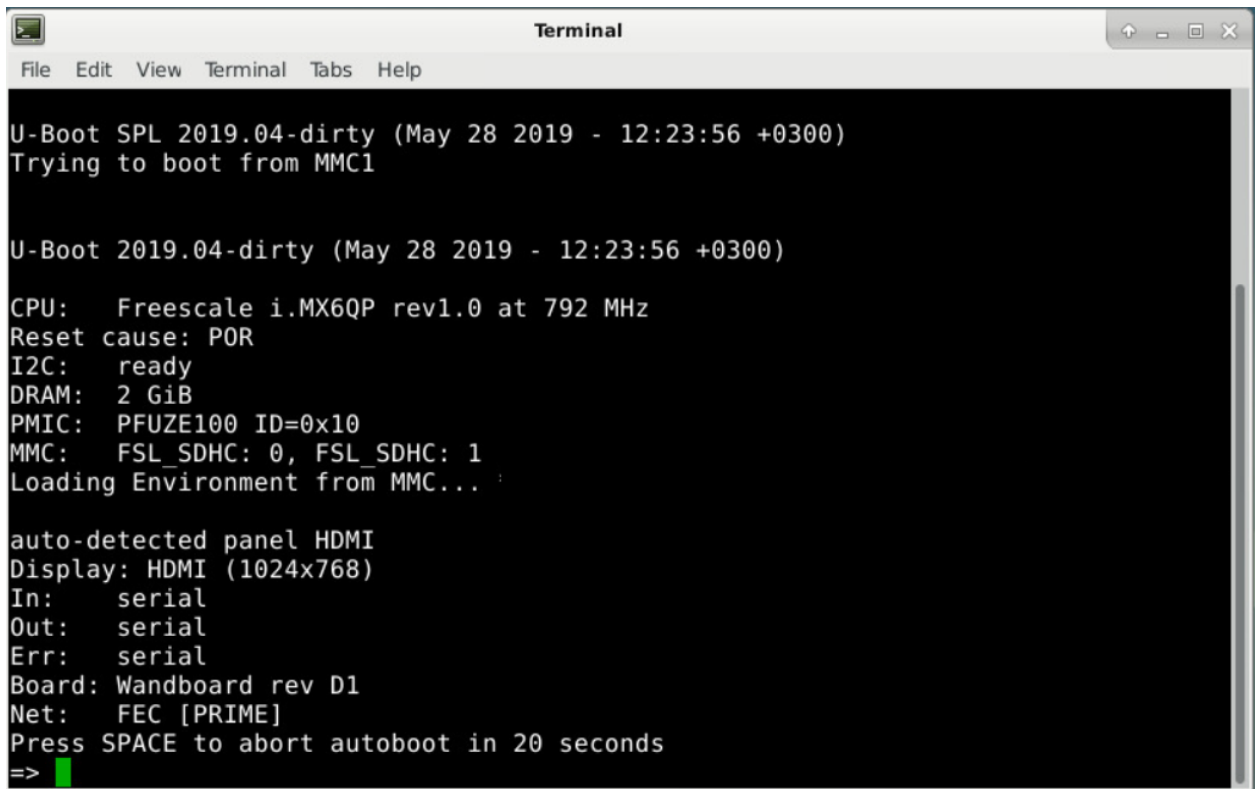
A screenshot of a terminal window titled "Terminal" with a menu bar containing "File", "Edit", "View", "Terminal", "Tabs", and "Help". The terminal output shows the U-Boot boot process. It starts with "U-Boot SPL 2019.04-dirty (May 28 2019 - 12:23:56 +0300)" and "Trying to boot from MMC1". This is followed by "U-Boot 2019.04-dirty (May 28 2019 - 12:23:56 +0300)". The output then lists hardware details: "CPU: Freescale i.MX6QP rev1.0 at 792 MHz", "Reset cause: POR", "I2C: ready", "DRAM: 2 GiB", "PMIC: PFUZE100 ID=0x10", and "MMC: FSL_SDHC: 0, FSL_SDHC: 1". It then says "Loading Environment from MMC...". Next, it reports "auto-detected panel HDMI", "Display: HDMI (1024x768)", "In: serial", "Out: serial", and "Err: serial". Hardware information includes "Board: Wandboard rev D1" and "Net: FEC [PRIME]". The prompt "Press SPACE to abort autoboot in 20 seconds" is shown, followed by the U-Boot prompt "=>" with a green cursor.

Fig. 4. Console application window with the ability to select the OS to boot to the target platform using the U-Boot software product.

In addition to being able to select the OS to boot into the target platform it is needed to have the ability to force the power on and off of the target platform from the control computer. Firstly, not every loaded OS contains software for rebooting the system. Secondly, software developers can make mistakes that lead to the complete inaccessibility of the target platform by software due to, for example, a loop or an emergency stop.

The USB device MP709 / RODOS-3 [14] (Fig. 5) manufactured by the Olymp company was chosen to provide a possibility of turning the power of the target platform on and off. This device connected to the USB port of the control computer allows software to obtain a power switch for controlling the power of the target platform. In other words it is possible to turn the power on and off programmatically without the need for direct physical access of the developer to the pushbutton switch of the target platform.



Fig. 5. USB relay / switch RODOS-3.

Thus, a scheme was created for full control of the target platform from the control computer without the need for the physical presence of the developer.

Consequently, it remains to resolve the issue of providing remote access to the control computer.

4.2. The choice of OS for the control computer

It was necessary to select an operating system for the control computer of the stand with the possibility of remote access to it by developers. Our initial tests using MS Windows as the OS for the control computer showed that this OS is not a convenient and an acceptable option for all developers. It does not satisfy all the requirements for the control computer listed above. The main reason is the lack of the possibility of simultaneous remote access of developers to the control computer via the Internet. The MS Windows Professional version supports remote desktop access via RDP (Remote Desktop Protocol) but only for one user and without the possibility of simultaneous access by several developers at once. Server versions of MS Windows support the simultaneous access of several clients to the desktop but they are quite an expensive solution due to the cost of the OS itself. In addition, the use of server versions of MS Windows imposes significant requirements to the hardware configuration of the control computer which leads to a significant increase in the costs of a stand.

The use of alternative software tools instead of the standard RDP protocol for MS Windows, for example, the well-known TeamViewer, also did not satisfy all the requirements for remote access to the control computer. It required significant additional costs to acquire licenses to use these software tools including for all participants in the development.

Therefore, Linux was chosen as the operating system for the stand control computer. It is a free solution, provides ample opportunities for remote management and support, and does not impose costly requirements on the hardware configuration of the computer. In addition, the Linux environment allows the installation of a huge number of additional software packages including everything necessary for the operation and management of the stand. This includes, for example, a TFTP server, a console application for interacting with the target platform, Python. Linux OS provides a flexible solution using all the necessary software tools on one personal computer without the need for additional hardware. At the same time this

computer can simultaneously perform other additional functions that are not related to the operation of the stand itself, such as a web server, a print server, a file server, etc.

4.3. Tools for remote access to the control computer

Linux OS provides remote access via the Internet to your desktop in various ways, allowing you to choose the most optimal solution for stand management. In addition, any version of Linux provides remote computer administration via the secure SSH (Secure Shell) protocol. This makes it possible to timely update the installed software products and eliminate vulnerabilities and errors found in the system, even without access to the computer's graphical interface. Linux also provides transmission and reception of files using an interception-protected protocol SCP (Secure Copy). At the same time, SSH / SCP clients are available for all common platforms providing the ability to administer even from the screen of a mobile smartphone.

One can install an RDP server for remote access to the graphical interface of the desktop and its windowed applications in Linux. Its usage is similar to the option with MS Windows. The Linux environment also supports the X11 protocol, i.e. remote launch of graphical applications over the network with output to the client computer screen when the X-Server program is running on this computer.

However, our tests have shown that these solutions do not provide the required level of security, Internet speed and usability. For example, the X11 protocol is open and unprotected from data interception. Then it becomes necessary to organize an additional secure channel, for example, a reverse SSH tunnel, in order to provide the necessary level of security. This would require significant effort on the part of everyone involved in the development to further customize their home computers. At the same time, the speed of displaying the graphical interface running on the application server on the client's screen is extremely low.

Finally we opted for the Virtual Network Computing (VNC) system [15] operating over the RFB (Remote FrameBuffer) protocol. This system provides sufficiently high-quality and high-speed remote access to the desktop of the control computer. At the same time, the VNC system is platform-independent, i.e. developers can access the toolbox from their home computers running Linux, MS Windows or MacOS without having to change their usual development environment. In addition, multiple clients (VNC viewer) can simultaneously connect to one VNC server. This satisfies the requirement for simultaneous remote access of several developers at once.

VNC and the RFB protocol do not use traffic encryption by default. So the question arises about ensuring the security of access to the stand and excluding the possibility of data interception. At the same time, it was highly desirable to eliminate the need to configure an encrypted tunnel by all users. Fortunately, it turns out that many VNC software developers are extending the standard RFB protocol to include their own encryption of the VNC server link. For example, a software implementation of VNC called TigerVNC [16] uses the OpenSSL cryptographic library [17] to securely encrypt connections, and the VNC session is encrypted including authentication and data transfer. Also, TigerVNC (VNC viewer) clients are available for all major platforms: for Linux, MS Windows, MacOS.

Thus, this choice fully meets all the criteria and requirements for a remote control workstation, i.e. for a control computer.

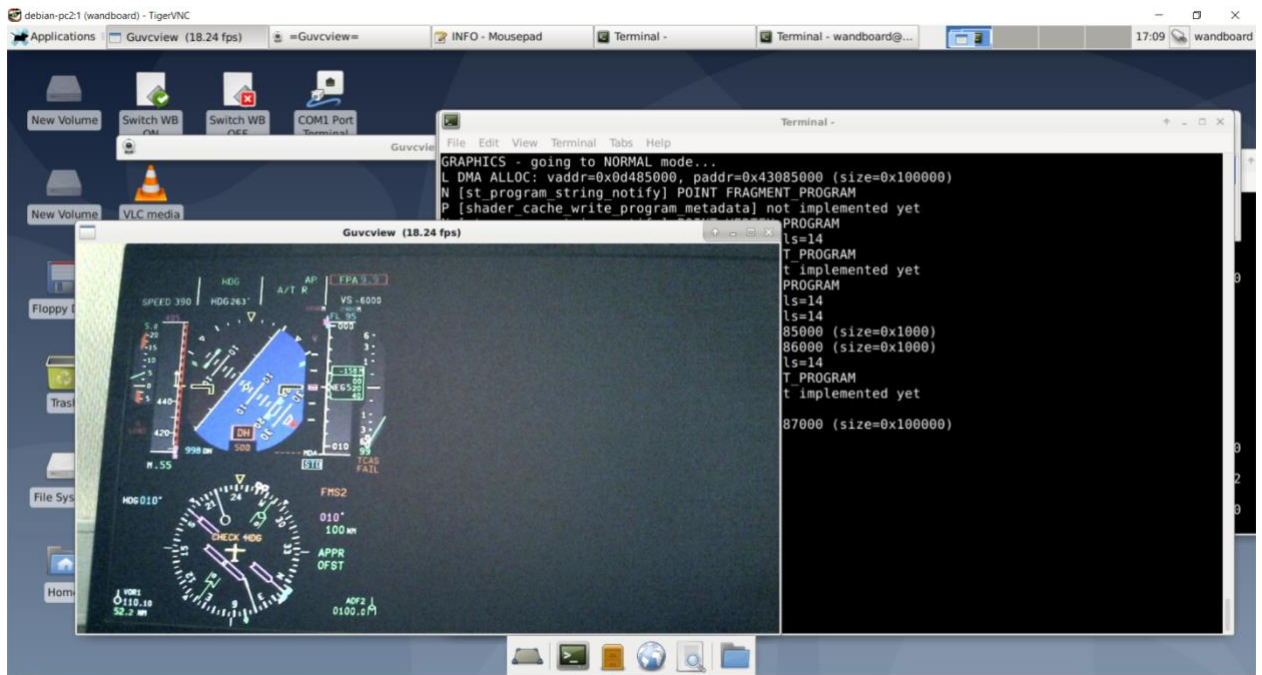


Fig. 6. Desktop of the control computer from the developer's client notebook via the TigerVNC viewer program. Windows with a view of the monitor of the target platform and the program execution protocol are open. Power on / off is implemented in the form of icons at the top left.

At the same time the additional efforts and time spent by all participants in the development are minimized and limited only to the installation of the VNC client program. The desktop view of the control computer of the stand from the developer's client machine via the TigerVNC viewer program is shown in Fig. 6. on the desktop, there is a window with a view of the target platform monitor obtained using a USB camera and the program execution protocol window. Power on and off are implemented as icons at the top left.

5. Visual observation of the results of the target platform

To organize visual observation of the target platform output, i.e. its display on the monitor screen, we used a completely natural solution, namely, observation with the help of cameras. Two types of cameras were used for this purpose: an IP camera with independent access via the Internet and a USB camera connected directly to the USB port of the control computer. The use of the first option (IP-camera) allows visual observation of the operation of the target platform on many different devices, even on the screen of a mobile smartphone, not only on the developer's home computer (Fig. 7). The image received from the USB camera is displayed on the remote desktop of the control computer (Fig. 6) and is available using the UVC (USB Video Class) system.

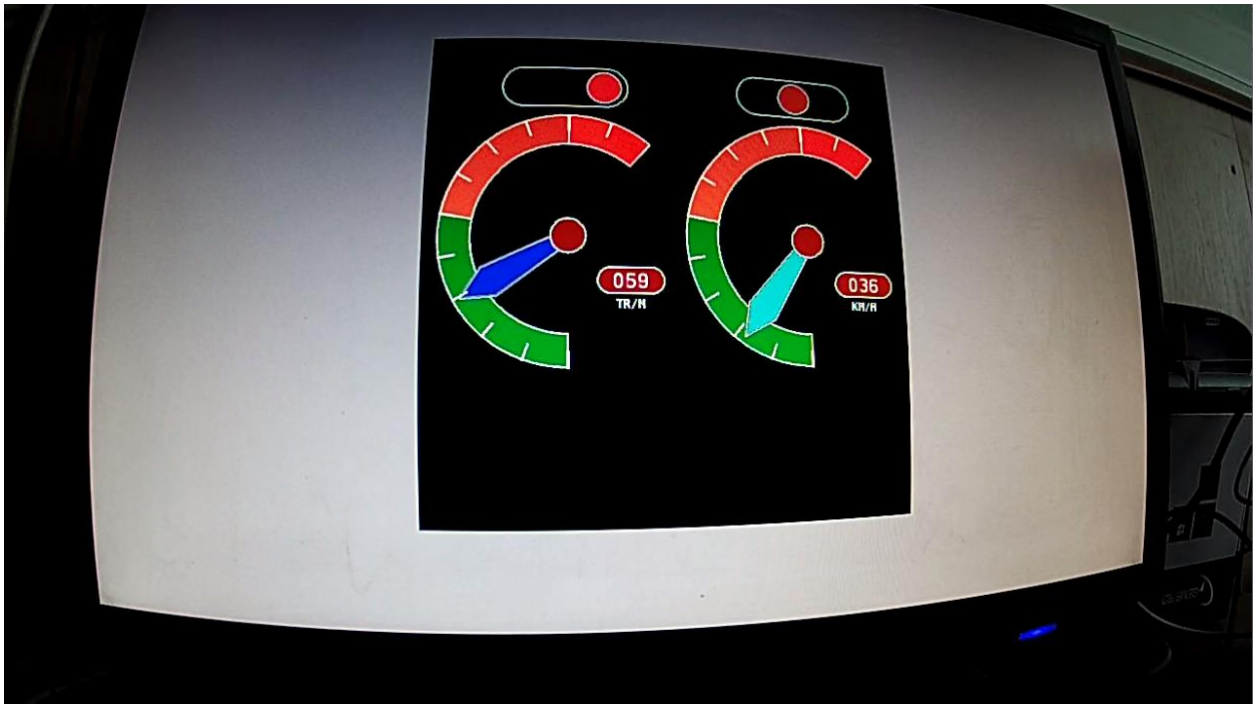


Fig. 7. Visual observation of the results of the test program running on the target platform using an IP camera.

Also, these methods allow you to record the results of visual observation in the form of a video file for further playback and demonstration to all participants in the development.

6. Conclusion

The stands for remote development of a visualization system for civil aircraft cockpit displays have been created and are successfully used around the clock at the Keldysh Institute of Applied Mathematics RAS. The development process is not interrupted even in the face of restricting workers' access to target platform equipment due to the worldwide pandemic.

The developed stand scheme is not highly specialized and can also be used for the development of other visualization systems that require remote access and control of equipment, providing the required level of safety and speed, as well as visual assessment of the results.

References

- [1] Safety critical visualization of the flight instruments and the environment for pilot cockpit. Scientific Visualization, 2021, 13(1), pp. 124 - 137, DOI: 10.26583/sv.13.1.09
- [2] J. L. Hardison, K. DeLong, P. H. Bailey and V. J. Harward, "Deploying interactive remote labs using the iLab Shared Architecture," 2008, 38th Annual Frontiers in Education Conference, 2008, pp. S2A-1-S2A-6 DOI: 10.1109/FIE.2008.4720536.
- [3] Matej Rabek, Katarina Zakova, "Scalable Remote Experiment Manager", IFAC-PapersOnLine, vol. 53, p. 17246, 2020.
- [4] Martin Kalúz, Pablo Orduña, Javier García-Zubia, Miroslav Fikar, L'uboš Ćirka, "Sharing Control Laboratories by Remote Laboratory Management System WebLab-Deusto", IFAC Proceedings Volumes, vol. 46, p. 345, 2013.
- [5] Ridha Ennetta, Ibrahim Nasri, Soufiene Bouall?gue, Thrasyvoulos Tsiatsos, Cyber-Physical Laboratories in Engineering and Science Education, p. 343, 2018.
- [6] Pablo Orduña, Luis Rodriguez-Gil, Javier Garcia-Zubia, Olga Dziabenko, Ignacio Angulo, Unai Hernandez, Esteban Azcuenaga, "Classifying online laboratories: Reality

simulation user perception and potential overlaps", Remote Engineering and Virtual Instrumentation (REV) 2016 13th International Conference, pp. 224-230, 2016.

[7] Laser lab with remote access. The Bauman University <http://bibl.laser.nsc.ru/download/laser-inform/472all-1.pdf>

[8] Internet-lab "Robototechnics". <http://fms.bmstu.ru/>

[9] Remote access to the CAD lab. https://miem.hse.ru/edu/ce/cadsystem/remote_access

[10] The remote access stands. <https://etu.ru/assets/files/nauka/razrabotki/Avtomatiz-Stendi-udal-dost.pdf>

[11] i.MX 6 Series Applications Processors. https://www.nxp.com/products/processors-and-microcontrollers/arm-processors/i-mx-applications-processors/i-mx-6-processors:IMX6X_SERIES Last accessed 05 Mar 2021

[12] B. Kh. Barladian, L. Z. Shapiro, K. A. Mallachiev, A. V. Khoroshilov, Yu. A. Solodelov, A. G. Voloboy, V. A. Galaktionov, I. V. Koverninskii. Visualization Component for the Aircraft Real-Time Operating System JetOS // Programming and Computer Software, 2020, 46(3), p. 167-175. DOI: 10.1134/S0361768820030020

[13] Das U-Boot -- the Universal Boot Loader <https://www.denx.de/wiki/U-Boot/WebHome>

[14] MP709 - USB relay <https://olimp-z.ru/mp709>

[15] Virtual Network Computing https://ru.wikipedia.org/wiki/Virtual_Network_Computing

[16] TigerVNC: Free, Lightweight, Fast and Reliable Remote Control / Remote Desktop Software <https://tigervnc.org/>

[17] OpenSSL. Cryptography and SSL/TLS Toolkit <https://openssl.org>