



RESEARCH ARTICLE

Virtual Remote Network Computing of User Appliances

Kaja Masthan¹, K. Sharath Kumar², V. Hari Prasad³

¹M.Tech (CSE), Sphoorthy Engineering College, JNTU, Hyderabad, India

²Associate Professor, Sphoorthy Engineering College, JNTU, Hyderabad, India

³Head of the Department (CSE & IT), Sphoorthy Engineering College, JNTU, Hyderabad, India

Abstract— Through the use of software VNC, acronym for virtual network computing, makes it possible to interact with a computer from any computer or mobile device on the Internet. VNC software provides cross-platform support allowing remote control between different types of computers. To use VNC you must have a network TCP/IP connection, a VNC server and a VNC viewer to connect to the computer running the VNC server. The open source version of VNC has been freely available since 1998, and more than 20 million copies of the software have been downloaded. The existing RFB protocol is extended straightforwardly to integrate video codecs. Next, the overall system architecture is modified from serial operation to parallel. Finally, we propose a modified region coding to further reduce the encoding time of screen images. The proposed methods are implemented into our prototype mobile VNC system, and practical performances are widely evaluated. We report that JPEG is the most suitable for mobile VNC in terms of both complexity and compression ratio. In addition, the proposed modified region coding can decrease encoding time and consequently increase screen update rate.

Key Terms: - mobile VNC; MJPEG; screen image coding; modified region coding

I. INTRODUCTION

VNC is a technology for remote desktop sharing. VNC enables the desktop display of one computer to be remotely viewed and controlled over a network connection. This technology is useful on home computers, allowing someone to access their desktops from another part of the house or while traveling. It is also useful for network administrators in business environments. VNC was created as an open source research project in the late 1990s. Since that time, several mainstream remote desktop solutions have been created based on VNC. The original development team produces the RealVNC package (see sidebar). Other popular derivatives include UltraVNC and TightVNC. VNC works similarly to the Remote Desktop application built into newer versions of Microsoft Windows. Unlike Windows Remote Desktop, VNC runs on older Windows computers, Linux/Unix and other non-Windows operating systems. VNC applications, however, are generally regarded as slower and offering fewer features and security options than Windows Remote Desktop.

It is also known As: virtual networking computing.



In this work, we propose a fast screen sharing method to improve screen update rate in mobile VNC systems. In case of mobile devices, high complexity video compression techniques cannot be employed due to their strict computation limit. However, the bandwidth limitation requires a certain level of compression ratio. Thus, there exists a trade-off between encoder complexity and compression ratio for fast mobile VNC systems. Virtual network computing (VNC) is a type of remote-control software that makes it possible to control another computer over a network connection. Keystrokes and mouse clicks are transmitted from one computer to another, allowing technical support staff to manage a desktop, server, or other networked device without being in the same physical location. VNC works on a client/server model: A VNC viewer (or client) is installed on the local computer and connects to the server component, which must be installed on the remote computer. The server transmits a duplicate of the remote computer's display screen to the viewer. It also interprets commands coming from the viewer and carries them out on the remote computer.

The remainder of the paper is organized as follows. Related works are described in Section II. In Section III, we give explanation on prototype system implementation. In Section IV, we describe video codec integration and protocol extension. Section V presents the proposed modified region detection methods for mobile VNC systems in detail. The experimental results of our prototype system are presented in Section VI.

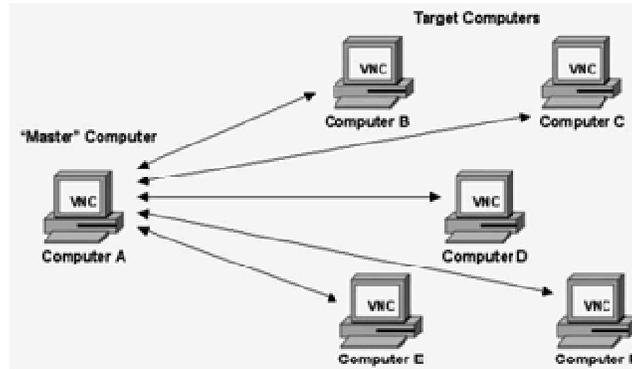
II. MOBILE VNC

VNC — Virtual Network Computing — allows to remotely view and control a given computer (the "Server") from one or more other computer(s) (the "Viewer(s)"). VNC was created at the Olivetti & Oracle Research Lab in the mid 1990's. The Server and the Client exchange via a well-defined protocol named RFB (Remote Frame Buffer.) While VNC is not new, it is widespread with several millions of users. The RFB protocol is light and its adaptability to various kinds of computers is outstanding. This is the reason why we found it well-suited aiming at another way to exchange visual data between mobile phones. The idea of porting VNC to mobile phones started in 2005 when I was attending a presentation of new services, provided by... a well renown mobile phone company. The speaker wanted to show what was happening on his mobile screen. He enjoyed to walk forth and back, gesticulating in front of the audience. Then, there was this fat TV cameraman bending over the speakers' shoulder, sweating and pesting at trying to catch the views from the mobile screen. This obviously didn't give the presentation the professionalism it deserved: people laughing, and a resulting image, displayed on a large screen, rather blurry. Back to my place, I looked on the Internet for VNC softwares ported on the mobile. I found VNC viewers, but, amazingly, no servers. So I decided to write one. RFB v4 protocol VNC Mobile Solution is based on an extended version of the standard RFB v3 protocol providing enhanced performance, security and the ability to implement vendor-specific protocol extensions.

VNC is platform independent and is compatible with any operating system. Computers must be networked with TCP/IP and have open ports allowing traffic from the IP addresses of devices that may need to connect. VNC was developed at AT&T Laboratories. The original VNC source code is open source under the GNU General Public License, and other variations are also available commercially.

III. THE RFB PROTOCOL

RFB ("remote framebuffer") is a simple protocol for remote access to graphical user interfaces that allows a client to view and control a window system on another computer. Because it works at the frame buffer level, RFB is applicable to all windowing systems and applications. This document describes the protocol used to communicate between an RFB client and RFB server. RFB is the protocol used in VNC.



The RFB protocol is a simple protocol for sending graphics to be displayed on a remote display (RFB stands for "remote framebuffer"). It is far simpler than X or ICA and therefore it places very little demand on the remote display in terms of processing power, memory, etc. The two endpoints in the RFB protocol are referred to, unsurprisingly, as the client and the server. The RFB client is the remote display. It simply takes rectangles of screen data with a given position and size and puts them into its framebuffer so that they appear in the correct place on the screen. The RFB server can be anything capable of producing a stream of such rectangles of screen data. In the dumbest implementation this could just repeatedly dump the whole of a framebuffer as fast as possible. This would be somewhat wasteful for a typical computer display where most of the screen is static for long periods. A better implementation sends updates only when changes to the screen are required, and only sends the areas of the screen which have changed. In our current implementation the RFB server consists of an X server with an extension which detects regions of the screen which have changed, plus an X application to actually send the appropriate rectangles from the X server's framebuffer. However the RFB protocol is at a low enough level that it is truly window-system independent. There is no reason why a Windows-based or Mac-based RFB server could not be provided. It is possible to hook the RFB server into a workstation's normal display, in which case the remote display is simply duplicating what is on the workstation's screen. More likely however is that the RFB server is not associated with an existing physical screen. Instead it has a "virtual framebuffer" - an area of ordinary memory treated like a framebuffer - where graphics and text are rendered by the windowing system and its applications.

In this case the remote display is the only physical screen showing the contents of the framebuffer. We are also likely to want to interact with the windowing system and its applications from the remote display. Keyboard and mouse/pen input is fed back to the windowing system by a simple addition to the RFB protocol.

Originally the RFB protocol was developed as a means of interacting with applications from an ATM network display device called the Videotile. The same basic RFB protocol is also used in the Virtual Network Computer.

IV. VIDEO CODEC INTEGRATION

The lossy codec based on Wavelet transform is frequently used in compression of natural image, motion picture and audio stream. But it isn't almost used in Remote Desktop protocol like VNC because classical lossless run-length codec like ZRLE has high performance in peculiar image which has big mono color region like desktop image. The lack of lossy codec cause low performance of motion picture in VNC. We introduce new codec "ZYWRLE(ZLib YUV Wavelet Run Length Encoding)" which applied wavelet transform partially and is based on ZRLE. In this new codec, network traffic reduced to 15%-50% traffic of ZRLE when we play motion picture. When sharing gaming or movie playbacks on a screen, low compression ratio leads to large amount of bit streams which cause network congestion and slow screen updates at the client. On the other hand, video encoding techniques such as MPEG exhibit high compression ratio but computation proportionally increases because of complex motion estimation and compensation. In particular, encoder complexity in the server is more critical for mobile devices. Thus, a method to encode screen

images should be carefully selected in terms of both encoding time and compression efficiency. As mentioned in the previous section, we integrated FFmpeg to our prototype system because it provides a variety of image and video encoders. Among them, we select MPEG2, MPEG4 and motion JPEG as candidate encoding methods, and their performances are reported in the experimental results.

V. JAVA: ACCESS THROUGH A BROWSER

When Sun Microsystems released the alpha version of the Java language and the HotJava browser in 1995, we realized we could implement the Videotile mechanism in Java to access applications through a Web browser. The thin-client paradigm made the adaptation to Java very straightforward. We wrote the original Java client in a day and the resulting class file was a mere 6 kilobytes in size. This eventually became the VNC applet described in more detail elsewhere.⁴ Any Java-capable browser could now provide access to a user's desktop, giving the mobility of the Teleporting system, but on a global scale.

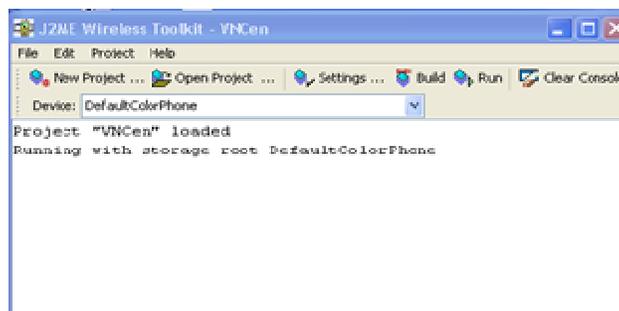
ANY USER INTERFACE, ANYWHERE

At ORL, we have used VNC to add mobility to workstation GUIs, where the concept of at least some form of remote inter-action is not new. But the protocol's simplicity could allow it to be used on a much wider range of hardware. Consumer electronics devices, such as CD players, usually have a highly specialized user interface and typically employ customized physical display devices. This has traditionally prevented such inter-faces from being mobile in the VNC sense of the word.

VI. VNC REDIRECTS

In computing, Virtual Network Computing (VNC) is a graphical desktop sharing system that uses the RFB Remote Frame Buffer protocol (remote framebuffer) to remotely control another computer. It transmits the keyboard and mouse events from one computer to another, relaying the graphical screen updates back in the other direction, over a network. VNC is platform-independent – a VNC viewer on one operating system may connect to a VNC server on the same or any other operating system. There are clients and servers for many GUI-based operating systems and for Java. Multiple clients may connect to a VNC server at the same time. Popular uses for this technology include remote technical support and accessing files on one's work computer from one's home computer, or vice versa.

VNC was originally developed at the Olivetti & Oracle Research Lab in Cambridge, United Kingdom. The original VNC source code and many modern derivatives are open source under the GNU General Public License. VNC in KDE 3.1 There are a number of variants of VNC[1] which offer their own particular functionality; e.g., some optimized for Microsoft Windows, or offering file transfer (not part of VNC proper), etc. Many are compatible (without their added features) with VNC proper in the sense that a viewer of one flavour can connect with a server of another; others are based on VNC code but not compatible with standard VNC.



VII. SAMPLE CODE

```
import javax.microedition.io.*;
import javax.microedition.lcdui.*;
import javax.microedition.midlet.MIDlet;
import javax.microedition.midlet.MIDletStateChangeException;
import javax.microedition.rms.RecordEnumeration;
import javax.microedition.rms.RecordStore;
import tk.wetnet.a.a;
import tk.wetnet.j2me.a.b;
```

```

// Referenced classes of package tk.wetnet.j2me.vnc:
//      a, b

public class VNC extends MIDlet implements CommandListener, Runnable
{

protected void _mthif()
{
    _fldfor++;
    _fldvoid.setValue(_fldfor);
}public void run()
{
    if(_fldelse.getString() == null || _fldelse.getString() != null &&
    _fldelse.getString().length() < 1)
    {
        Alert alert = new Alert("No Host!", "The host box is blank\nSo many problems it will cause\nPlease fill it in",
null, AlertType.ERROR);
        Alert _tmp = alert; alert.setTimeout(-2); s.setCurrent(alert, p); return;
    }
    if(_fldif.getString() != null && _fldif.getString().length() < 6 &&
    _fldif.getString().length() != 0)
    {
        Alert alert1 = new Alert("Password Problem", "Password is too short\nGreater than six letter it must\nPlease
write some more.", null,
AlertType.ERROR); Alert _tmp1 = alert1; alert1.setTimeout(-2); s.setCurrent(alert1, p);
return;
    }
}
}

```

VIII. FUTURE WORK

We are now building VNC software for a variety of desktop platforms, but it would not be difficult to make remote access practical for a wider range of devices. We can envisage cheap hardware that might, for example, drive a 7-segment LCD and also emit a VNC equivalent over a USB or RS232 link. The VNC commands to draw and erase each segment could be stored as a sequence of bytes in a small amount of ROM and sent over a communications link when the segment is lit or switched off. Hardware such as this, if made in quantity, could be very cheap and could allow for mobility of much more than just a conventional “desktop.” If built into television sets, VNC viewers could allow them to act as displays for a very wide range of devices—including, of course, the PC at the office.

IX. CONCLUSION

VNC is cross-platform, allowing remote control between different types of computer. For ultimate simplicity, there is even a Java viewer, so that any desktop can be controlled remotely. VNC has a wide range of applications including system administration, IT support and helpdesks. It can also be used to support the mobile user, both for hot desking within the enterprise and also to provide remote access at home, or on the road. The system allows several connections to the same desktop, providing an invaluable tool for collaborative or shared working in the workplace or classroom, we found that MJPEG is the most suitable for mobile VNC systems in terms of both complexity and compression ratio. Besides, the proposed modified region coding can further decrease encoding time, and consequently increase screen update rate at the client. Various practical and diverse experimental results demonstrate that the proposed methods guarantee fast screen image encoding without visual quality degradation. In particular, modified region detection is significantly effective for gaming video contents with small texture regions.

REFERENCES

- [1] T. Richardson, Q. Stafford-Fraser, K. Wood, and A. Hopper, “Virtual network computing,” *IEEE Internet Computing*, vol. 2, no. 1, pp. 33–38, Jan./Feb.1998.
- [2] P. M. Corcoran, F. Papal, and A. Zoldi, “User interface technologies for home appliances and networks,” *IEEE Trans. Consumer Electron.*, vol.44, no. 3, pp. 679-685, Aug. 1998.
- [3] K. Tsunashima, T. Shida, H. Kawano, T. Sato, and H. Kosaka, “Compact programmable network display system for portable projectors,” *IEEE Trans. Consumer Electron.*, vol. 55, no. 2, pp.

- 312-315, May 2009.
- [4] D. Thommes, Q. Wang, A. Gerlicher, and C. Grecos, "RemoteUI: A high-performance remote user interface system for mobile consumer electronics devices," Proc. of IEEE International Conference on Consumer Electronics (ICCE 2012), pp. 670-671, Jan. 2012.
 - [5] P. M. Corcoran, F. Papal, and A. Zoldi, "User interface technologies for home appliances and networks," IEEE Trans. Consumer Electron., vol.44, no. 3, pp. 679-685, Aug. 1998.
 - [6] K. V. Kaplinsky, "VNC tight encoder-data compression for VNC," Proc. of the 7th International Scientific and Practical Conference of Students, Post-graduates and Young Scientists (MTT 2001), pp. 155-157, Feb.2001.
 - [7] K. J. Tan, "A remote thin client system for real time multimedia streaming over VNC," Proc. of IEEE International Conference on Multimedia and Expo (ICME 2010), pp. 992-997, July 2010.
 - [8] X. Zhang and H. Takahashi, "A hybrid data compression scheme for improved VNC," Systemics, Cybernetics and Informatics, Vol. 5, No. 2 pp. 1-4, 2007.
 - [9] H. Shen, "A high-performance remote computing platform," Proc. of IEEE International Conference on Pervasive Computing and Communication (PerCom 2009), pp. 1-6, Mar. 2009.
 - [10] S. Rao, H. Vin, and A. Tarafdar, "Comparative evaluation of server-push and client-pull architectures for multimedia servers," Proc. of the 6th International Workshop Network and Operating System Support for Digital Audio and Video (NOSSDAV), pp. 45-48, Apr. 1996.
 - [11] C. Taylor and J. Pasquale, "Improving video performance in VNC under high latency conditions," Proc. of the International Symposium on Collaborative Technologies and Systems (CTS 2010), pp.26-35, May 2010.
 - [12] W. J. Kim, K. Cho, and K. S. Chung, "Stage-based frame-partitioned parallelization of H.264/AVC decoding," IEEE Trans. Consumer Electron., vol. 56, no. 2, pp. 1088-1096, May 2010.
 - [13] Sheng Liang, Java Native Interface: Programmer's Guide and Specification, 1st ed., Prentice Hall, 1999.
 - [14] Jae-Hyeok Lee, Ha-Young Ko, Jong-Ok Kim, "Fast modified region detection for mobile VNC systems," Proc. of International Conference on Awareness Science and Technology (iCAST 2012), Aug. 2012.

Authors Bibliography

Mr. Kaja Masthan is pursuing M.Tech (CSE) in Sphoorthy Engineering College, JNTU Hyderabad.

Mr. K. Sharath Kumar (PhD) Associate professor in Sphoorthy Engineering College, JNTU Hyderabad.

Mr. V.Hariprasad working as Head of the Department (CSE & IT) in Sphoorthy Engineering College, Hyderabad. He is pursuing PhD in JNTU Kakinada.