Development of the Communication Interface for the Cognitive Model of Complicated Human-Computer Interaction Task

Xianliang Mu, Shaoyao Zhang, Yu Tian, Lifen Tan, Chunhui Wang[a]

National Key Laboratory of Human Factors Engineering, China Astronaut Research and Training Center, Beijing, 100094, China

[a]email:chunhui_89@163.com

Keywords: Real-Time Communication; Adaptive Control of Thought-Rational; User Datagram Protocol; Multicast; Rendezvous and Docking

Abstract. The cognitive modeling of the HCI task has been a hot area in HCI research. And it is an important method to understand the cognitive decision process and mechanism behind operators' behavior. Since the software of HCI task and the software of cognitive modeling generally run on different platforms, it is critical to develop the communication interface between different platforms, especially in the cognitive modeling research of the simulated manual RVD task. To realize a real-time interaction between the cognitive model developed by Common Lisp language and the task simulator developed by VC++ language, this paper developed the communication interface between these two platforms. And the interface is developed based on UDP and the multicast technique. This paper proposed a solution with the foreign-function interface feature of Common Lisp Language to solve the problem that Common Lisp Language doesn't support the multicast technique directly. Test results showed that the communication interface could support data sending and receiving and fulfill the requirement of the real-time communication.

Introduction

Manual RVD (Rendezvous and Docking) task is a complicated human-computer interaction task for astronauts. Also this task is a critical research content for space ergonomics and long-duration space flight in the future [1] [2]. Baozhi Wang, Guohua Jiang, Jiangang Chao (ET al.) designed and realized a semi-physical ergonomics experimental system for the purpose of studying ergonomics problem in manned RVD task. This platform accurately simulated the RVD task in dynamics and could automatically generate behavior and performance data of human [3].

Recently, the method called cognitive modelling becomes more and more popular in the research of cognitive science [4]. In this method, a calculation model is established to evaluate the cognitive theory and phenomenon. Compared with traditional experimentation, cognitive modeling costs much less and can predict task performance, workload and probable fault quantitatively, which is a new way to evaluate the design of human-computer interface. For this reason, we built a cognitive model for RVD task and this model was established based on the cognitive architecture of Adaptive Control of Thought-Rational (ACT-R) [5]. The cognitive architecture is an expatiation for the architecture of brain at the abstract level, which explains how the brain think [6].

In order to evaluate and verify the cognitive model, we should guarantee that the cognitive model and the subjects work at the same environment. So the cognitive model should communicate with the simulator in real time. The method of communication is usually based on network. Internationally, people usually take the method of UDP (User Datagram Protocol) [7, 8].

For the purpose of realizing real-time communication between cognitive model and simulator [9], we developed the interface between them. And the UDP can satisfy the requirement of real-time interaction. Compared with TCP (Transmission Control Protocol), the UDP is faster in the transmission speed and costs less to run in the system. However, the UDP is less reliable in the data transmission. UDP supports unicast, broadcast and multicast. Multicast means sending datagram to a mainframe unit in the network. And a mainframe can join in one or several multicast unit [10]. So multicast satisfies the requirement of dynamics and can save network bandwidth to reduce the workload of network.

The architecture of ACT-R is established based on Common Lisp language. Unfortunately, the ACT-R architecture doesn't supply the network communication interface. So we have to realize the communication protocol based on Common Lisp language. In this paper, Allegro Common Lisp was chosen to realize the communication protocol. It supports the TCP and UDP, but can't support multicast directly [11].

In this paper, we took use of the characteristic of supporting Foreign-Function Interface (FFI) in Common Lisp language to realize the UDP multicast protocol. Using C++ language to realize the UDP multicast protocol has been a mature technology, so it is not a key point in our paper.

Overall scheme

In this simulated Manual RVD task, cognitive model simulates the astronaut to do the manual RVD task. The simulator stands for the task environment. In terms of communication interface, the cognitive model based on ACT-R architecture is a client and the simulator is a server. The overall scheme is shown in Fig. 1.

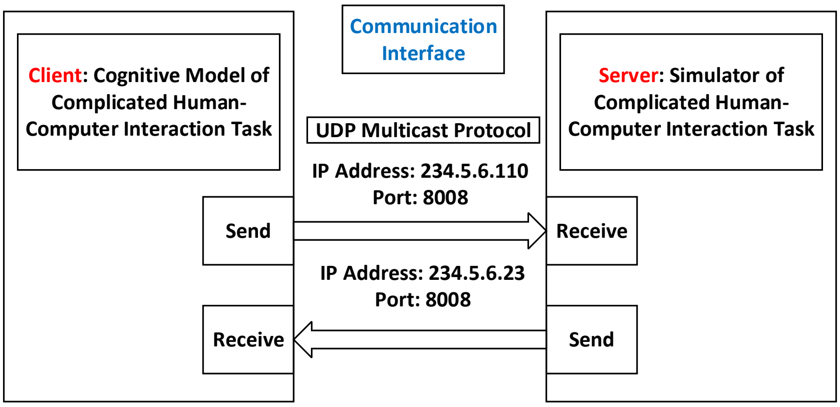

Communication Interface
Client: Cognitive Model of Complicated Human-Computer Interaction Task
UDP Multicast Protocol
Server: Simulator of Complicated Human-Computer Interaction Task
IP Address: 234.5.6.110
Port: 8008
Send
Receive
IP Address: 234.5.6.23
Port: 8008
Receive
Send

Fig.1. Overall scheme of the communication interface

In this scheme, two multicast units were established. And the unit-1's address is "234.5.6.110" with a port of "8008", which is used to send message to server. While, the unit-2's address is "234.5.6.23" with a port of "8008", which is used to send message to client. The workflow of the system is followed: (1) client receives the real-time task information from server; (2) the cognitive model simulates human's cognitive behavior and send out control commands; (3) the sever receive control commands from client and change the real-time task information based on the commands; (4) circulate from (1) to (3) until the task completes.

In the case of sever, UDP multicast communication protocol was realized based on Winsock 2 using C++ language [12]. As a mature technology, this part won't be repeated here. Now, we talk about how to establish UDP multicast communication protocol through Common Lisp language.

The realization of client's communication interface

The cognitive model based on ACT-R is developed by Common Lisp language. Common Lisp supports the characteristic of Foreign-Function Interface (FFI). FFI allows the running Common Lisp program to load code file from foreign language to realize the function of foreign language.

Common Lisp can't support multicast directly, so this paper attends to realize the client's UDP multicast protocol using the characteristic of FFI in Common Lisp. The protocol includes the UDP sending function and UDP receiving function. Firstly, realize the UDP sending function and UDP receiving function based on Winsock 2. Then, compile the source files to DLL (Dynamic Link Library) files. And with that, load these two DLL files into Common Lisp code. In the end, define corresponding Common Lisp function using FFI. As a result, we can call corresponding functions in Common Lisp.

The client run in the Windows 7 system. The development environment of cognitive model is

Allegro Common Lisp 9.0 and the development environment of C++ is Microsoft Visual Studio 2010. The corresponding Winsock 2 functions are shown in Table 1.

Table 1 Introductions of the Winsocks 2 functions used

Name	Description
bind	Bind a socket at an address.
closesocket	Close an existing socket.
recvfrom	Receive a datagram and save the source address.
sendto	Send data to specified destination address.
setsockopt	Set options of a socket.
socket	New a specified type of socket.
WSACleanup	Stop to use the DLL file, Ws2_32.DLL
WSAStartup	Initiate to use DLL file, Ws2_32.DLL

3.1 UDP multicast sending function

The flow chart of UDP multicast sending function based on Winsock 2 is shown in Fig. 2 (a).

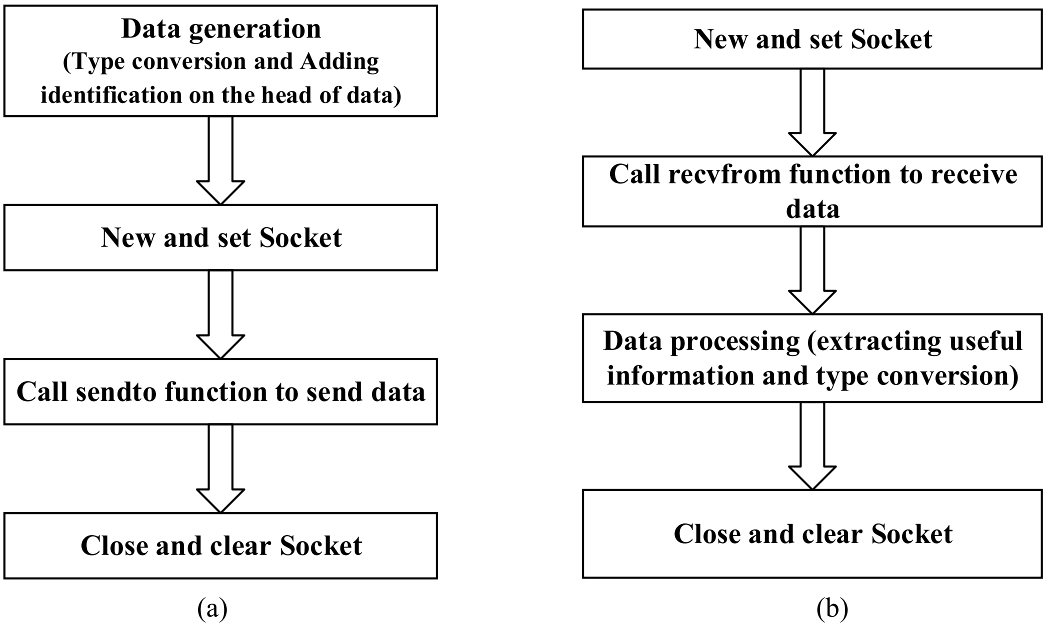

Data generation
(Type conversion and Adding identification on the head of data)
New and set Socket
Call sendto function to send data
Close and clear Socket
(a)
New and set Socket
Call recvfrom function to receive data
Data processing (extracting useful information and type conversion)
Close and clear Socket
(b)

Fig. 2. Flow charts of two functions: (a) flow chart of UDP multicast sending function; (b) flow chart of UDP multicast receiving function

Data generation means changing the 6 DoF (Degree of Freedom) control instructions of cognitive model into char data that can be sent by UDP protocol. And there are two progress: conversion of data from float to char and add identification on the head of char data. Then, call WSAStartup function and socket function to new a socket. And with that, call setsocktopt function to complete the setting of address allowed to be reused. Once again, call sendto function to send char data to multicast address 1 (234.5.6.100, port 8008). In the end, call closesocket function to close socket and call WSACleanup function to clear data.

The statement of sending function is: *int sendFloatArray (float original_control_out[6])*. The formal parameter stands for the output of 6 DoF control instructions which is generated from cognitive model.

UDP multicast receiving function

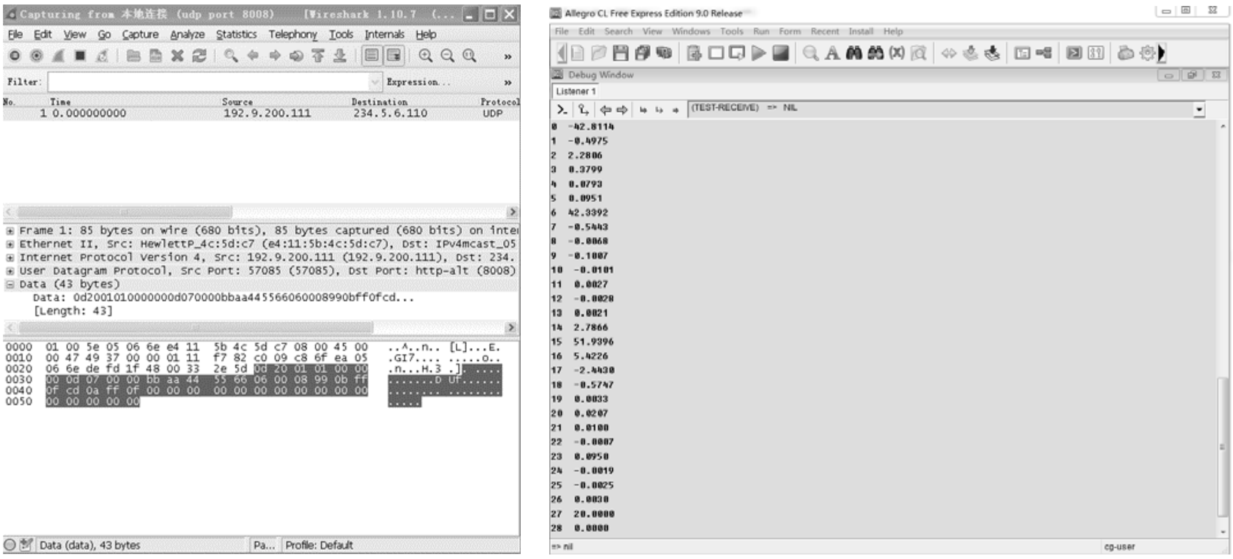
As shown in Fig. 2 (b), it is the flow chart of UDP multicast receiving function based on Winsock 2. Firstly, call WSAStartup and socket functions to new a socket. Secondly, call setsockopt function twice to complete the setting of address allowed to be reused and join in the setting of multicast. Thirdly, call bind function to bind the new socket to multicast address 2 (234.5.6.23, port 8008). And with that, call recvfrom function to receive the char data sent from multicast address 2 in the server. Then conduct data processing to delete the identification on the head of char data and extract the useful data to converse the char data into float data. In the end, close the socket and clear the data.

Test results of the communication interface

The server is a simulator that run in a computer with the operating system of Windows XP, while the client is a cognitive model that run in a computer with the system of Windows 7. These two computers are connected through reticle. The test includes client sending test, client receiving test and real-time running test.

Client sending test

The cognitive model, as a client, will sent the 6 DoF data to simulator. In the server computer, we use the software of Wireshark 1.10.7 to conduct network capture. The results are "0d 20 01 01 00 00 00 0d 07 00 00 bb aa 44 55 66 06 00 08 99 0b ff 0f cd 0a ff 0f 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00" which are expressed in hexadecimal, as shown in the Fig. 3 (a). The useful data is "66 06 00 08 99 0b ff 0f cd 0a ff 0f", which is the same with the data captured by C++ program. The results show that the client multicast sending function completes the data conversion and sending.


Capturing from 本地连接 (udp port 8008) [Wireshark 1.10.7 (...
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression...
Data (data), 43 bytes
Pa...
Profile: Default
Allegro CL Free Express Edition 9.0 Release
File Edit Search View Windows Tools Run Form Recent Install Help
Debug Window
Listener 1
cg-user

```
No.  Time           Source           Destination   Protocol
   1 0.000000000    192.9.200.111    234.5.6.110   UDP

⊞ Frame 1: 85 bytes on wire (680 bits), 85 bytes captured (680 bits) on inte
⊞ Ethernet II, Src: HewlettP_4c:5d:c7 (e4:11:5b:4c:5d:c7), Dst: IPv4mcast_05
⊞ Internet Protocol Version 4, Src: 192.9.200.111 (192.9.200.111), Dst: 234.
⊞ User Datagram Protocol, Src Port: 57085 (57085), Dst Port: http-alt (8008)
⊟ Data (43 bytes)
    Data: 0d2001010000000d070000bbaa4455660600089 90bff0fcd...
    [Length: 43]

0000  01 00 5e 05 06 6e e4 11  5b 4c 5d c7 08 00 45 00   ..^..n.. [L]...E.
0010  00 47 49 37 00 00 01 11  f7 82 c0 09 c8 6f ea 05   .GI7.... .....o..
0020  06 6e de fd 1f 48 00 33  2e 5d 0d 20 01 01 00 00   .n...H.3 .]. ....
0030  00 0d 07 00 00 bb aa 44  55 66 06 00 08 99 0b ff   .......D Uf......
0040  0f cd 0a ff 0f 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0050  00 00 00 00 00                                      .....
```

(a)

```
(TEST-RECEIVE) => NIL
0  -42.8114
1  -0.4975
2  2.2806
3  0.3799
4  0.0793
5  0.0951
6  42.3392
7  -0.5443
8  -0.0068
9  -0.1007
10  -0.0101
11  0.0027
12  -0.0020
13  0.0021
14  2.7866
15  51.9396
16  5.4226
17  -2.4430
18  -0.5747
19  0.0033
20  0.0207
21  0.0100
22  -0.0007
23  0.0950
24  -0.0019
25  -0.0025
26  0.0030
27  20.0000
28  0.0000
=> nil
```

(b)

Fig.3 (a) The data server received from client; (b) the useful data client received from server

Client receiving test

The server sends real-time information to multicast address 2 (234.5.6.23, port 8008) continuously. And the client receives data and prints out the results. As shown in Fig. 3 (b), the client multicast receiving function has received the data sent from server successfully and completed the extraction of useful data and conversion of data type.

Real-time running test

The server runs the simulated program and the client runs cognitive model. The cognitive model will call communication interface function to communicate with simulated program. The running results showed that our communication interface could satisfied the requirement of real-time interaction.

Conclusion and expectation

In this paper, in order to solve the problem that Common Lisp can't support the multicast directly, we developed a communication interface based on UDP multicast protocol to realize the real-time interaction between cognitive model and simulator in the task of simulated Manual RVD. We have compiled the multicast sending and receiving functions into DLL files, loading these files into Common Lisp and defining corresponding functions. The test results show that our communication interface could satisfied the requirement of real-time interaction.

The solution in this paper can be contributed to all kinds of cognitive modeling work of complicated human-computer interaction tasks.

Acknowledgement

This study was supported by the Foundation of Key Laboratory of Science and Technology for National Defense (No. 9140A26070215KG57417, No. 9140C770102140C77313), the Feitian Foundation of China Astronaut Research and Training Center (No. FTKY201505), the Foundation of National Key Laboratory of Human Factors Engineering (No. SYFD140051802), and the Experimental Technique Youth Fund Project (No. SYFD1400618).

References

[1] Shanguang CHEN, Guohua JIANG, Chunhui WANG. Advancement in Space Human Factors Engineering [J]. Manned Spaceflight, 2015 21(2) 95-105.

[2] Shanguang CHEN, Chunhui WANG, Xiaoping CHEN, Guohua JIANG. Study on Changes of Human Performance Capabilities in Long-duration Spaceflight [J]. Space Medicine & Medical Engineering, 2015 28(1) 1-10.

[3] Baozhi WANG, Guohua JIANG, Jiangang CHAO, ET al. Design and Implement of Manned Rendezvous and Docking Ergonomics Experimental System [J]. Space Medicine & Medical Engineering, 2011 24(1) 30-35.

[4] Fum, D., F.D. Missier, A. Stocco. The Cognitive Modeling of Human Behavior: Why a Model is (Sometimes) Better than 10,000 Words [J]. Cognitive Systems Research, 2007 8(3) 135-142.

[5] Anderson, J.R., D. Bothell, M.D. Byrne, ET al. An Integrated Theory of the Mind [J]. Psychological review, 2004 111(4) 1036-1060.

[6] Anderson, J.J.R., C.J. Lebiere. The Atomic Components of Thought [M]. Psychology Press, 1998.

[7] Byrne, M.D., A. Kirlik. Using Computational Cognitive Modeling to Diagnose Possible Sources of Aviation Error [J]. The international journal of aviation psychology, 2005 15(2) 135-155.

[8] Dimperio, E., G. Gunzelmann, J. Harris. An Initial Eevaluation of a Cognitive Model of UAV Reconnaissance. Proceedings of the Seventeenth Conference on Behavior Representation in Modeling and Simulation [C]. 2008 165-173.

[9] Shaoyao ZHANG, Shanguang CHEN, Chunhui WANG, ET al. An Integrated Cognitive Model of Manual Rendezvous and Docking Task Based on ACT-R Cognitive Architecture [J]. Space Medicine & Medical Engineering, 2015 28(2) 109-116.

[10] W.Richard Stevens. TCP/IP Detailed Annotation Roll 1 [M]. China Machine Press, 2005.

[11] Peter Seibel. Practical Common Lisp Programming [M]. Press and Telecom Press, 2011.

[12] Haimin SUN. Master Windows Sockets Network Development-Realization Based on Visual C++ [M]. Press and Telecom Press, 2008.