

# 三种基本网络传送模式的编程实现与分析

成卫青<sup>1,2</sup>, 王雪梅<sup>1,2</sup>, 豆仁福<sup>3</sup>, 周宁宁<sup>1,2</sup>

(1. 南京邮电大学 通达学院, 江苏 南京 210003;

2. 南京邮电大学 计算机学院, 江苏 南京 210003;

3. 95856 部队, 江苏 南京 210028)

**摘 要:**为顺应社会对网络人才的需求,很多学校已经将“网络编程”作为计算机专业或网络工程专业必修的实践课程或课程设计。教学过程中发现套接字编程令不少同学望而生畏。因此,文中以引导学生快速入门为目的,简要阐述了三种基本网络传送模式单播、组播和广播的编程实现思路和要点,包括基于 TCP 的服务程序和客户端程序,基于 UDP 的单播、组播和广播收发程序的编写,及多线程的实现框架等。此外,探讨了结合网络监听工具的网络编程调试方法。在课程设计中尝试先向学生介绍采用基本网络传送模式的应用程序的编写,再鼓励学生采用文中的调试方法独立编写稍稍复杂的小程序,发现不少同学已经能够较好地完成课程设计任务。实践表明掌握基本网络传送模式编程是网络编程入门的关键。

**关键词:**单播;组播;广播;服务程序;网络编程

**中图分类号:**TP31

**文献标识码:**A

**文章编号:**1673-629X(2011)07-0132-06

## Three Basic Transfer Mode Network Programming and Analysis

CHENG Wei-qing<sup>1,2</sup>, WANG Xue-mei<sup>1,2</sup>, DOU Ren-fu<sup>3</sup>, ZHOU Ning-ning<sup>1,2</sup>

(1. Tongda College, Nanjing University of Posts and Telecomm., Nanjing 210003, China;

2. School of Computer Sci. & Tech., Nanjing Univ. of Posts and Telecomm.,  
Nanjing 210003, China;

3. PLA 95856, Nanjing 210028, China)

**Abstract:** To meet the demand of the society on network talents, many universities have already taken "network programming" as a compulsory course or curriculum design for computer speciality or network engineering speciality. In the teaching process, found that the socket programming is formidable for many students. Thus, for the purpose of quick start guide for students, it briefly describes programming ideas and key points for the three basic network transmission modes: unicast, multicast, and broadcast, including the network programming of TCP-based servers and clients, UDP-based unicast, multicast and broadcast senders and receivers, as well as multi-threaded implementation framework. In addition, a debugging method of network programming combined with network monitoring tools is discussed. During a curriculum design, tried to introduce the programming about basic network transmission modes to students, and then encouraged them to implement slightly complex programs independently using the debugging method. It was found that many students have been able to better complete the task of curriculum design. Practice shows that learning the programming about basic network transmission modes is the key for learning network programming.

**Key words:** unicast; multicast; broadcast; server programs; network programming;

## 0 引 言

经过多年的计算机网络课程教学和网络编程课程设计指导,发现愿意探索网络编程的同学越来越少了。究其原因,估计与学生为了未来就业而疲于获得一个个证书有关,网络、软件工程师、英语六级、CCIE 等证

书成了学生的追求目标,他们越来越缺乏探求稍显困难的网络编程领域奥秘的兴趣。而实际上随着网络应用渗透到各行各业,国内很多企业还是非常需要擅长网络编程的人才的。此外,根据培养目标,计算机相关专业的本科生应当掌握一定的网络编程技能<sup>[1-7]</sup>。文中旨在通过分析比较三种基本网络传送模式的实现,引导更多的学生轻松地初步掌握网络编程基础知识,真正做到入门,为进一步的学习树立信心、增强兴趣。

TCP/IP 网络支持三种基本网络传送模式:单播、组播和广播。其中单播分为基于 TCP 的单播和基于

收稿日期:2010-12-17;修回日期:2011-03-25

基金项目:中兴通讯研究基金(nj2009 09080002);南京邮电大学教学改革研究资助项目(JG00409JX08, JG00409JX09)

作者简介:成卫青(1972-),副教授,博士,CCF 会员,研究方向为计算机网络测量。

UDP的单播,组播和广播只能基于UDP。目前单播最常用,很多应用程序都基于单播,例如Web应用、FTP应用、Email应用、即时通讯应用等。组播有利于节约网络资源,视频会议、网络授课、网络TV等应用可以基于组播实现。广播不太常用。

## 1 三种基本网络传送模式实现

文中以Windows Sockets<sup>[8]</sup>为例,阐述基于套接字实现具有基本网络传送模式应用程序所涉及的要点。Windows Sockets是一套开放的、支持多种协议的Windows下的网络编程接口,由头文件和动态链接库Winsock 2 DLL(WS2\_32.DLL)组成<sup>[8]</sup>。基于套接字的网络应用程序开发涉及的公共要点包括:

(1)在应用程序中用编译指令:#pragma comment(lib,"ws2\_32"),将Winsock库文件ws2\_32.lib加入到工程中,另外必须包含Winsock头文件winsock.h。

(2)应用程序首先应调用WSAStartup函数告诉Winsock DLL应用程序所请求的版本,如果请求的版本大于等于Winsock DLL支持的最低版本,则调用成功,并以请求的版本和DLL支持的最高版本二者中的较小值作为提供给应用程序的DLL版本。有关WSAStartup的调用和容错处理在VC6.0的帮助文档MSDN中有直接可用的例程。

WSAStartup成功调用之后,应用程序才可以调用其他Windows Sockets函数。终止使用WS2\_32.DLL的服务时,应用程序必须调用WSACleanup以便WS2\_32.DLL释放分配给应用程序的所有资源。

下面分传送模式介绍各自其他的实现要点。

### 1.1 基于TCP的单播

基于TCP的单播应用程序又分为提供服务的服务器程序和主动请求服务的客户程序。

#### 1.1.1 TCP 服务器程序

支持多个客户方的TCP服务器程序的实现主要包括如下步骤:

##### (1)创建流式套接字。

套接字是一个通信端点、一个对象,通过它应用程序收发经过网络的数据分组。套接字有类型之分,并且与进程相关联。套接字分为流式套接字、数据报套接字和原始套接字三种。流式套接字基于TCP提供一个保证有序无重复交付的双向数据流。使用socket()函数<sup>[8]</sup>,其中类型参数设为SOCK\_STREAM创建一个流式套接字(这里也称为初始套接字)。创建的套接字默认处于阻塞模式(blocking mode),处于阻塞模式的套接字的阻塞函数默认在完成操作返回之前什么也不能做。文中的示例全部使用阻塞模式。对于阻塞模式,可以用setsockopt()<sup>[8]</sup>指定接收超时时间(SOL\_

SOCKET的SO\_RCVTIMEO选项),让recv()函数不停地进行读动作,直到接收到用户报文,或超时时间到。将套接字设置为非阻塞方式需要调用ioctlsocket():

```
// If iMode = 0, blocking is enabled; If Mode != 0, non-blocking mode is enabled.
```

```
//The ioctlsocket function controls the I/O mode of a socket.
```

```
u_long iMode = 1;
```

```
ioctlsocket(m_socket, FIONBIO, &iMode); //设置套接字使用非阻塞模式
```

(2)将本地地址和端口与套接字绑定。

使用bind()函数将本地地址和服务器使用的TCP端口号与初始套接字绑定。本地地址可以使用全0地址(INADDR\_ANY),也可以获取本地单播地址。获取本地单播地址的方法是:调用gethostname()获得主机名,再调用gethostbyname()根据主机名获得主机信息,其中包括主机的IPv4地址。

(3)设置套接字处于监听状态。

调用listen()函数使初始套接字处于监听进入连接请求的状态,也即使服务进程处于被动连接状态。处于监听状态的流套接字将维护一个客户连接请求队列。

(4)循环地在初始套接字上接受新的连接请求,并为新连接创建一个新套接字和一个专门处理该连接的线程,线程中处理数据传输和连接释放。

调用accept()函数接受排在客户连接请求队列之首的连接请求,该函数创建一个新的与初始套接字具有相同绑定的套接字,与客户套接字建立TCP连接通道,并返回主动连接实体(客户方)的IP地址和使用的TCP端口号,以及新套接字的句柄。新创建的套接字处理实际的连接,不接受其他连接请求,而初始套接字一直保持打开状态并监听新的连接请求。客户服务器之间的实际连接建立在accept()所创建的套接字上。此外,可以为每个已接受的连接专门创建一个线程处理该连接上的数据传输和连接释放,这部分程序大致如下:

```
while(1){
    sock1 = accept(s, (struct sockaddr FAR *)
    &ClientAddr,&Addrlen); //s为初始套接字
    accept 失败处理.....
    int ThreadID; DWORD thread;
    //创建线程
    ThreadID = (int) CreateThread(NULL, 0,
    (LPTHREAD_START_ROUTINE)
    .(SpecifiedforEachClient), &sock1, 0, &thread); //
    如果成功,则返回为 1
```

```

if(! ThreadID) {
printf("线程创建失败! \n");continue;
}
}

```

在线程函数中,注意必须使各线程分别拥有自己的内存空间存放用户数据,否则并发线程接收的数据会发生覆盖冲突。线程函数框架如下:

```

int SpecifiedforEachClient(LPVOID lpPara)
{
SOCKET sock_client = * (int *) lpPara; //lp-
Para 指向 sock1
char rcvdata[ MAX_RCVSIZE]; //存放接收到的数据
int rcvlen;
while(1) { //这里仅是示例,实际流程取决于具体应用
.....
if(( rcvlen = recv ( sock_client, rcvdata, MAX_
RCVSIZE,0))<=0) { //接收
printf("recv failed! \n"); //客户端关闭套接字
时接收失败
closesocket ( sock_client); return -1;
}
//处理接收到的用户数据,其中可以调用 send()
向客户方发送信息,
//根据情况调用 shutdown() 停止接收、发送或收
发用户数据
.....
}
return 1;
}

```

以上是基于动态连接库 WS2\_32. DLL,利用 Win-Sock API 函数实现 TCP 服务程序的要点。此外,还可以利用 VC++ MFC 类库中提供的套接字类,例如 CAsyncSocket、Csocket 等较方便地实现网络编程。该文是基础,有了这个基础,学习更复杂或高级的网络编程<sup>[9~11]</sup>更容易入门。

### 1.1.2 TCP 客户程序

主动请求服务的客户程序的实现包括如下步骤:

#### (1) 创建客户套接字。

与创建服务器套接字一样,语句为:sock = socket (AF\_INET, SOCK\_STREAM, 0)。

#### (2) 连接服务器。

调用 connect() 函数使客户套接字与服务器套接字建立连接,函数参数中指定客户套接字以及服务器的 IP 地址和 TCP 端口。此外对于未绑定的客户套接

字(尚未调用 bind() 函数),调用 connect() 函数后系统会将套接字与本地地址和一个自由端口号相绑定。

#### (3) 数据传输和连接释放。

根据实际需要调用 send() 和 recv() 发送和接收数据,调用 shutdown() 关闭连接,具体流程取决于具体的应用。另外,如果客户程序同时打开多个套接字,则对每个套接字最好单独创建一个线程处理,否则视具体情况而定。

#### (4) 关闭套接字。

调用 closesocket (sock); 关闭套接字。

### 1.1.3 数据传送实现与分析

流式套接字使用 TCP 协议收发数据。应用程序和 TCP 之间的交互基本原理如下:

#### (1) 发送过程。

应用程序调用 send() 将用户数据→套接字发送缓存并通知 TCP 输出进程 tcpout<sup>[12]</sup>;

tcpout 将套接字发送缓存中的数据按照一定的算法构建一个个 TCP 报文段发给 IP 进程。

send() 函数声明为: int send( \_in SOCKET s, \_in const char \* buf, \_in int len, \_in int flags)。设置不同 len 调用 send(), 使用网络监听工具观察实际产生的 TCP 报文段,并进行分析比较,有助于理解上述过程。

#### (2) 接收过程。

· IP 进程收到以本机为目的的 TCP 报文段,会通知 TCP 输入进程 tcpinp 处理;

· tcpinp 分析 TCP 报文段首部并做处理(如发确认等),包括根据目的端口号确定套接字,再将报文段中的数据存入套接字接收缓存。注意由通信两端的 TCP 负责数据传送的可靠性,保证按顺序提交已通过校验的数据;

· 应用程序调用 recv() 从套接字接收缓存中读入用户数据,但能读入多少与设置的 recv 参数和接收缓存中当前保存的数据量有关。因此调用 recv() 一定要获取返回参数以了解实际读入的字节数。

本例的套接字采用阻塞模式,并且使用默认的 TCP 流量控制算法 Nagle 算法,会发现发送出去的用户数据块大小和接收到的用户数据块大小并不一定一一对应。

对于有些应用,例如基于 TCP 发送语音包,可能希望 TCP 遵从应用程序安排数据发送,这时可以设置关闭 Nagle 算法。方法是调用 setsockopt() 设置套接字选项 TCP\_NODELAY:

```

DWORD tcp_nodelay = 1;
setsockopt( sock, IPPROTO_TCP, TCP_NODELAY,
(char *) &tcp_nodelay, sizeof( DWORD));

```

setsockopt() 函数声明为: int setsockopt( \_in SOCKET s, \_in int level, \_in int optname, \_in const char \* optval, \_in int optlen)。常用的套接字选项见表1。

表1 套接字选项的 level 和 optname  
(setsockopt 函数的第2,3参数)

level	套接字选项适用级别说明	optname 举例
SOL_SOCKET	Socket options applicable	SO_RCVTIMEO, SO_SNDBTIMEO, SO_BROADCAST
IPPROTO_IP	at the IPv4 level.	IP_TTL, IP_MULTICAST_IF, IP_MULTICAST_TTL, IP_MULTICAST_LOOP, IP_ADD_MEMBERSHIP, IP_DROP_MEMBERSHIP
IPPROTO_TCP	at the TCP level.	TCP_NODELAY
IPPROTO_UDP	at the UDP level.	UDP_NOCHECKSUM

## 1.2 基于 UDP 的单播

UDP 是无连接的,一个基于 UDP 的单播应用程序可分为提供服务的服务程序和主动请求服务的客户程序,也可以合二为一。下面介绍兼有服务器和客户功能的 UDP 单播程序(对等程序)的编写方法。允许多个对等方访问又能主动访问别的对等方的 UDP 单播程序的实现主要包括如下步骤:

### (1) 创建数据报套接字。

数据报套接字支持面向记录的数据流,不保证数据被交付、有序接收以及没有重复。使用 socket() 函数,其中类型参数设为 SOCK\_DGRAM 创建数据报套接字。创建的套接字默认处于阻塞模式。

### (2) 将本地地址和端口与套接字绑定。

使用 bind() 函数将本地地址和要使用的 UDP 端口号与套接字绑定。本地地址可以使用全 0 地址(INADDR\_ANY)。

(3) 创建一个线程用于循环地在套接字上接收用户数据并根据需要做相应的处理。

实现这一功能的函数如果是成员函数,核心代码如下:

```
bool CComm::createRecvThread( sockaddr_in &srval )
{
    srv = srval; //srv 是数据成员
    RecvSocket = socket( PF_INET, SOCK_DGRAM, 0 );
    if ( RecvSocket == INVALID_SOCKET ) {
        printf( "Error: RecvSocket 创建失败\n" ); fflush( 0 ); return false;
    }
    if ( bind( RecvSocket, ( struct sockaddr * ) &srval, sizeof( srv ) ) != 0 ) {
```

```
printf( "Error: 绑定失败\n" ); fflush( 0 ); close-
socket( ListenSocket ); return false;
}
```

//创建线程

```
int ThreadID; DWORD thread;
```

```
ThreadID = ( int ) CreateThread ( NULL, 0,
( LPTHREAD_START_ROUTINE ) ( CComm:: ListenThread ), ( void * ) this, 0, &thread ); //CComm::
ListenThread() 是线程函数
```

if( ! ThreadID ) { // ThreadID 如果为 0, 则线程创建失败

```
printf( "线程创建失败\n" ); return false;
}
```

```
return true;
}
```

在 void \* CComm::ListenThread( void \* data ) 中调用 recvfrom() 函数接收用户数据并获悉对等方的 IP 地址和端口号。

(4) 主线程可以根据需要发送数据报。

一个数据报套接字既可以作为服务器,接收来自任何客户(对等方)的请求;也可以作为客户,调用 sendto() 主动向任何服务器(对等方)发起通信。

## 1.3 基于 UDP 的组播

IP 组播基于 UDP,组播应用程序包括一个发送者和若干接收者,接收者是属于一个组播组的若干成员。因此,一个组播包的源地址是发送者的单播地址,而目的地址是组地址。发送者可以不是组播组成员,也可以加入组播组。发送者和接收者都需要创建数据报套接字,而且发送者和接收者其实都可以基于各自创建的套接字,进行双向通信。为易于理解,下面假设发送者仅发送,接收者仅接收,编程要点如下。

### 1.3.1 组播发送者

能够发送组播包的 UDP 应用程序的实现主要包括如下步骤:

(1) 创建数据报套接字 ssock。

(2) 设置外出组播包的 TTL 值。

调用 setsockopt( ssock, IPPROTO\_IP, IP\_MULTICAST\_TTL, ( char \* ) &tttl, sizeof( ttl ) ) 设置组播 IP 包的生存时间(TTL)。如果组播组成员都在本地物理网络上,则 TTL 应设置为 1。数据报套接字默认用于组播的 TTL 即为 1。

(3) 将本地地址和端口与套接字绑定(可选)。

可以使用 bind() 函数将本地地址和发送组播包的 UDP 端口号与套接字绑定,不是必需的,若不绑定则 ssock 使用随机端口号。本地地址获取方法:调用 gethostname() 获得主机名,再调用 gethostbyname() 获

得已知主机名的主机信息,其中包括主机的 IPv4 单播地址。

(4) 设置组播接收者的套接字地址及发送组播包给组播接收者。

首先要设置组播接收者的套接字地址,其中 IP 地址为一组播地址,端口号为组播接收者使用的 UDP 端口号。然后调用 `sendto()` 函数即可向组播接收者发送信息。

可以将套接字加入组播组,这是可选的,加入方法见下一小节。此外,基于套接字 `ssock` 还可以接收信息,如果 `ssock` 加入了组,则可以接收组播包,否则只能接收单播包。

(5) 关闭套接字。

### 1.3.2 组播接收者

能够接收组播包的 UDP 应用程序的实现主要包括如下步骤:

(1) 创建接收数据报套接字 `rsock`。

(2) 将本地地址和端口与套接字绑定。

使用 `bind()` 函数将某个本地 IPv4 单播地址 A 和接收组播包的 UDP 端口号与套接字绑定,这是必须的。本地地址获取方法同上小节。

(3) 将套接字加入组播组 `mcaddr`。

先要定义一个组播地址,再使用 `IPPROTO_IP` 级别的 `IP_ADD_MEMBERSHIP` 选项调用 `setsockopt()` 设置套接字 `rsock`,将本地接口(地址为 A)加入该组播组,关键代码如下:

```
struct ip_mreq mreq;
memcpy(&(mreq.imr_interface), localaddr, sizeof(
struct in_addr)); //本地地址
memcpy(&(mreq.imr_multiaddr), mcaddr, sizeof(
struct in_addr)); //组播地址
int ret = setsockopt(rsock, IPPROTO_IP, IP_ADD_
MEMBERSHIP, (char *)&mreq, sizeof(mreq));
```

(4) 接收组播包。

调用 `recvfrom()` 在 `rsock` 上接收组播发送的数据,还可以获悉发送者的 IP 地址和端口号。注意,基于 `rsock` 还可以发送组播包(TTL 默认值为 1,可以设置其他值)和单播包。

(5) 关闭套接字。

## 1.4 基于 UDP 的广播

### 1.4.1 广播发送者

能够发送广播包的 UDP 应用程序的实现主要包括如下步骤:

(1) 创建数据报套接字 `ssock`。

(2) 设置套接字用于发送广播数据。

调用 `setsockopt()` 设置:

```
BOOL optval = TRUE;
```

```
setsockopt(ssock, SOL_SOCKET, SO_BROADCAST, (char *)&optval, sizeof(optval));
```

(3) 设置广播接收者的套接字地址及向广播接收者发送广播包。

首先要设置广播接收者的套接字地址,其中 IP 地址为一广播地址,可以是受限广播地址(全 1),也可以是指向子网的广播地址,如 10.10.8.255,端口号为广播接收者使用的 UDP 端口号 P。然后调用 `sendto()` 函数即可向广播接收者发送信息。

(4) 关闭套接字。

### 1.4.2 广播接收者

能够接收广播包的 UDP 应用程序的实现主要包括如下步骤:

(1) 创建接收数据报套接字 `rsock`。

(2) 将本地地址和端口与套接字绑定。

使用 `bind()` 函数将某个本地 IPv4 单播地址和接收广播包的 UDP 端口号与套接字绑定,这是必须的。注意并不是将套接字与 IP 广播地址绑定。

(3) 接收组播包。

调用 `recvfrom()` 在 `rsock` 上接收广播发送的数据,还可以获悉发送者的 IP 地址和端口号。

(4) 关闭套接字。

## 2 网络程序调试

网络程序的调试除了需要传统的程序调试方法,如设置断点,查看(watch)变量、表达式和内存的值,控制执行(继续运行、单步运行和运行到指定光标处)等,至少还需要一个网络监听工具辅助测试与调试。利用网络监听工具,可以很方便地核实程序是不是如希望的那样发送 IP 包,例如检查源 IP 地址、目的 IP 地址、源端口、目的端口,以及用户数据或应用层协议数据单元是否正确;了解了发送情况,对于接收程序的编写与调试也是必要的。例如,由于有网络序和主机序之分,编程时很容易写错端口号:

```
sockaddr_in addr; addr.sin_family = AF_INET;
addr.sin_port = 8888;
addr.sin_addr.s_addr = inet_addr("10.10.8.255");
```

再向 `addr` 发送 UDP 包;

用监听工具监听程序的运行会发现根本没有目的端口为 8888 的 UDP 包,有的是目的端口为 47138 的 UDP 包。这时查阅资料容易知道端口号的设置应改为:

```
addr.sin_port = htons((u_short)8888); //主机序
向网络序转换。
```

### 3 结束语

文中以引导学生快速初步掌握网络编程方法为目的,简明扼要地阐述了三种基本网络传送模式单播、组播和广播的编程实现思路和要点以及多线程的实现框架。此外,初步探讨了结合网络监听工具的网络程序调试方法和程序实际数据传送情况分析方法。

目前已经在课程设计中尝试先向学生介绍采用基本网络传送模式的应用程序的编写,然后再鼓励学生发挥主观能动性借鉴文中的数据传送分析和调试方法自己编写稍稍复杂的小程序,例如基于 TCP 的大文件传输工具、基于 UDP 的多方文字聊天工具等,发现不少同学已经不再惧怕网络编程,能够较好地完成课程设计任务。

#### 参考文献:

- [1] 王西锋,张晓李. 网络编程能力培养模式的探索与实践[J]. 计算机教育, 2009(2):93-94.
- [2] 成卫青,杨哲睿. 网络编程实验设计与教学研究[J]. 实验科学与技术, 2010, 8(2):99-101.
- [3] 刘 森,刘怀亮. 计算机专业《网络编程》实验教学改革探索[J]. 实验室科学, 2007(2):25-27.
- [4] 刘 琰,常 斌,罗军勇,等. 面向能力培养的网络编程技术课程教学方法探讨[J]. 计算机教育, 2010(18):52-

55.

- [5] 李向丽,李 磊,陈 静. 网络实验仿真与网络技术实践[J]. 计算机技术与发展, 2006, 16(3):74-76.
- [6] 李 鹏. 网络编程技术课程的教学改革思路[J]. 西安邮电学院学报, 2010, 15(2):166-168.
- [7] 张晓明,杜天苍,秦彩云. 计算机网络编程课程的教学改革与实践[J]. 实验技术与管理, 2010, 27(2):4-7.
- [8] Microsoft. socket Function[EB/OL]. 2011-01-20. [http://msdn.microsoft.com/en-us/library/ms740506\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/ms740506(v=vs.85).aspx).
- [9] Malhotra A, Sharma V, Gandhi P, et al. UDP based chat application[C]//2nd International Conference on Computer Engineering and Technology (ICCET). [s. l.]:[s. n.], 2010: 374-377.
- [10] Law K L E, Leung R. A design and implementation of active network socket programming [C]//Eleventh International Conference on Computer Communications and Networks. [s. l.]:[s. n.], 2002: 78-83.
- [11] Juszkievicz K. UNIX Network Programming, Volume 1: The Sockets Networking [J]. IEEE Communications Magazine, 2004, 42(5):20-21.
- [12] 史蒂文,科 默. 用 TCP/IP 进行网际互连:设计、实现与内核(ANSI C 版)(第2卷)[M]. 第3版. 张 娟,王 海,黄述真,译. 北京:电子工业出版社, 2008.

(上接第131页)

要低,在小型文档图像库中,根据特征对文档图像库进行分类,建立特征库索引,可以大大提高查找效率,实现文档图像间特征相似度的快速比较。

### 4 结 论

结合了行列方向投影特征对文档图像进行了相似度比较,其模型具有稳定性,当文档图像比例发生缩放、倾斜都有一定的鲁棒性。实验表明,在不同的 K 均值分类次数的情况下,综合利用段落特征和行特征进行 K 均值分类查找比顺序查找进行特征比较其效果要好,且随着迭代次数的增加,文档图像特征比较效率得到了提高。相似度度量的研究对下一步的文档图像检索提供了重要的参考依据。如何针对非文本文档图像的特征提取和相似度比较以及如何对中、大型文档图像特征库进行文档图像间的快速比较实现快速检索,也需要做更深入的研究以及提出更加灵活和方便的索引结构。

#### 参考文献:

- [1] Liu Hong, Feng Suoqian, Zha Hongbin, et al. Document Image Retrieval Based on Density Distribution Feature and Key Block Feature [C]//Proceedings of the 2005 Eight Interna-

tional Conference on Document Analysis and Recognition. [s. l.]: IEEE, 2005.

- [2] 胡芝兰,林行刚,严 洪. 基于分层密度特征的文档图像检索[J]. 清华大学学报, 2006, 46(7):1231-1234.
- [3] 赵 珊,郑清洁. 基于字符串匹配技术的图像检索算法[J]. 高技术通讯, 2010, 20(2):117-120.
- [4] 孙 远,周刚慧,赵立初,等. 灰度图像匹配的快速算法[J]. 上海交通大学学报, 2000, 34(5):702-704.
- [5] 罗钟铨,刘成明. 灰度图像匹配的快速算法[J]. 计算机辅助设计与图形学学报, 2005, 17(5): 967-969.
- [6] Herrmann P, Schlageter G. Retrieval of document images using layout knowledge [C]//In Proc. 2nd ICDAR. [s. l.]:[s. n.], 1993:537-540.
- [7] 阳方林. 一种新的快速图像匹配算法[J]. 计算机工程与应用, 2005(5):51-52.
- [8] 王佐林,王希常,刘 江. 基于数学形态学的文档图像段落标记及其应用[J]. 山东师范大学学报(自然科学版), 2007, 22(4):27-29.
- [9] 张 田,王希常,陈昌华. 基于特征的文档图像检索[J]. 计算机工程, 2009, 35(22):176-178.
- [10] 刘宝生,闫莉萍. 几种经典相似性度量的比较研究[J]. 计算机应用研究, 2006(11):1-3.
- [11] 陈卫兵. 几种图像相似性度量的匹配性能比较[J]. 计算机应用, 2010, 30(1): 98-99