LXI 网络仪器拓扑图生成的设计与实现

刘永斌,李宥谋,王 涛,赵成青
(西安邮电大学 计算机学院,陕西 西安 710000)

摘 要:研究了 LXI 总线下的网络仪器实时拓扑图的动态生成、刷新以及对拓扑图上节点进行各种管理,属于 LXI 网络仪器管理系统中监管平台的一部分。通过监管平台,可以将网络仪器拓扑直观地呈现并在此基础上远程、实时地对网络仪器进行测控、监管与维护,形成对 LXI 网络仪器的远程和统一管理,克服了时空和地域的限制。LXI 网管系统中除监管模块外,还包括后台和代理模块,在得到网络仪器拓扑或对拓扑上的仪器设备进行 SNMP 的 GET、SET 操作时,就需要后台模块来完成监管模块与代理模块之间的间接信息交互。网络仪器的拓扑结构在这里有两级,由局域网内连接的代理以及各代理不同接口连接的不同测量仪器形成,代理即拥有 uC/OS-II 和 Linux OS 的 ARM 系列开发板,测量仪器包括传感器、示波器和函数信号发生器等。

关键词:LXI;实时;拓扑图;SNMP;监管;后台;代理
中图分类号:TP315 **文献标识码**:A **文章编号**:1673-629X(2018)11-0150-05
doi:10.3969/j.issn.1673-629X.2018.11.033

Design and Implementation of LXI Network Instrument Topology Generation

LIU Yong-bin,LI You-mou,WANG Tao,ZHAO Cheng-qing
(School of Computer Science,Xi'an University of Posts and Telecommunications,Xi'an 710000,China)

Abstract:We study the dynamic generation and refreshing of real-time topology of network instruments under the LXI bus and various management of the nodes on the topology map,which are all part of the regulatory platform of LXI network instrument management system. Through the regulatory platform,network instrument topology can be visually presented and on the basis the network instrument can be measured,controlled and maintained remotely and in real-time,formation of LXI instrument remote and unified management,overcoming the limit of time and space. In addition to the supervisory module,the LXI network management system also includes the background and agent modules. When the network instruments topology is acquire,or the SNMP GET or SET operation is performed on the topological devices,the background module is required to complete the indirect information exchange between supervisory module and the proxy module. The topological structure of the network instrument has two levels here. It is formed by different measuring instruments connected by the agents connected in the LAN and different interfaces of the agent. The agent is the ARM series development board with uC/OS-II and Linux OS. The measuring instruments include sensors,oscilloscope and function signal generator.

Key words:LXI;real-time;topology;SNMP;supervisor;background;proxy

0 引 言

随着网络技术及信息技术的快速发展,仪器仪表的测控技术领域发生了巨大变化,出现了以 LXI 总线为基础的测控技术。LXI 网络仪器是一种结构化、模块化的仪器系统,仪器中各模块之间、仪器与网络之间均通过 LXI 总线进行互联,这样可不受机箱和零槽控制器的制约,使得集成更方便,也使得仪器中各测量模块相互独立,具备模块化特点,突破了传统通信技术的时空和地域限制。而且通过代理使得不带网口的测量仪器网络化,使其大大超越了传统测试测量仪器的通讯性能及局限性。

而对 LXI 网络仪器的测控、监管与维护的前提是监管平台应提供良好的网络仪器拓扑图,通过良好的拓扑图既可对代理及其所接网络仪器提供直观的链接

收稿日期:2017-11-24 **修回日期**:2018-03-22 **网络出版时间**:2018-05-28
基金项目:陕西省重大科技创新专项资助项目(2010ZKC02-08);嵌入式 LXI 网络仪器开发及产业化(2015KTCQ01-04)
作者简介:刘永斌(1993-),男,硕士研究生,研究方向为嵌入式系统开发与设计;李宥谋,教授,研究方向为集成电路设计、嵌入式系统开发与设计。

和分布等情况,也可提供操作简单、功能完善的测量和管理服务。因此 LXI 网络仪器拓扑图的发现、呈现与管理显得尤为重要。文中就拓扑图的动态生成、刷新以及对拓扑图上节点的状态和事件展开研究,并进行设计与实现。

1　拓扑图生成的设计与实现

1.1　数据库的设计

整个 LXI 网络仪器[1]管理系统的数据库管理统一由监管平台处理,该系统的数据信息存储在名为 xnmsdb 的 mysql5.5.54 版本数据库中,其中包含的数据表是根据监管平台所提供的功能确定的,而网络仪器拓扑管理作为监管平台的一部分,涉及的关键数据表有代理编号与其 MAC 地址映射表(t_AgentIDMAC)和节点信息及拓扑关系表(t_RunningTreeView)。由于篇幅有限,只给出这两张表的结构设计。

t_AgentIDMAC 的结构设计:

```
CREATE TABLE `t_AgentIDMAC` (
`id`int(11) NOT NULL,//代理编号,非空
`mac` varchar(20) NOT NULL,    //代理 MAC 地址,非空,字符串类型
PRIMARY KEY (`id`),      //id 为主键
UNIQUE KEY `mac` (`mac`)      //唯一性约束,防止 MAC 地址重复
)
```

t_RunningTreeView 的结构设计:

```
CREATE TABLE `t_RunningTreeView` (
`id`int(11) NOT NULL AUTO_INCREMENT,  //节点唯一标识号,自增
`name`varchar(20) NOT NULL,           //节点名称,非空
`pid` int(11) NOT NULL, //本节点的父节点 ID,非空
`ip` varchar(20) DEFAULT NULL,        //代理 IP 地址
`mac` varchar(20) DEFAULT NULL, //代理 MAC 地址
`dscr` varchar(100) DEFAULT NULL, //节点信息描述
`interface`int(2) DEFAULT NULL, //代理的接口号
PRIMARY KEY (`id`) //id 字段为主键
)
```

1.2　拓扑图结构的布局设计

该监管平台开发环境是使用 Windows OS 中 Microsoft Visual Studio 2012 的 IDE,利用 C#[2]语言在.NET Framework4.5 平台下进行开发。在 C#的窗体应用程序开发时,要求在监管平台的主控界面上显示局域网中的网络仪器拓扑图[3]。拓扑是一种层次关系,而在面向组件的 C#中,窗体开发时它本身提供的显示节点层级关系的控件只有 TreeView 控件,基于纵向的目录树结构,相当于书的目录样式,虽然功能完善,但是结构单一,不直观,不灵活,不能够适应特定的需求。所以对网络仪器拓扑图的管理需要有更符合人机操作关系的特定的拓扑图,那么就需要利用 C#提供的图形设备接口 GDI+[4]来绘制出更符合需求的拓扑结构。

拓扑图布局结构采用树状结构,整个 LXI 网络仪器管理系统最多有 32 个代理,每个代理最多可接 4 个测量仪器,根据实际情况可分为 8 组显示具体的拓扑图,每组可通过按钮分别显示。每组都为三级拓扑结构,根节点作为第一级,没有实际意义,起到连接代理的作用,代理为第二层节点,测量仪器为第三级节点,即叶子节点,连接在代理下。如果实际局域网中设备不能形成 8 组,那么只有存在的组的按钮才具备显示该组拓扑的事件。绘制拓扑图的关键是要把每组的三级拓扑层次结构绘制出来,并在拓扑周围绘制出必要的简易文字说明,绘制过程的控制中代码细节很多,这里只给出 GDI+操作的主要步骤,对于每一组拓扑采用自上而下、从左到右的画法,如下:

(1)Graphics graphic = panel1.CreateGraphics();即在窗体 form 中的 panel1 面板上创建画板,接下来就可以在 panel1 上绘制图形、线条、文字等,但绘制的前提是要根据 panel1 的大小以及三级拓扑层次结构的大小及位置确定合理的 X、Y 坐标,确定绘制对象的长度、粗细、形状、颜色等属性。

(2)根据坐标和绘制对象的属性,利用 Pen 类实例化画笔对象,绘制根节点,形状为圆形,并利用上述的 graphic[5]对象在圆形中绘制一个 Image 图形作为根节点标志;再根据根节点相对 panel1 位置绘制出相应的线条,用于连接四个代理。

(3)根据坐标和绘制对象属性,利用 Pen 类实例化画笔对象,通过 for 循环从左到右绘制出四个矩形,用来作为填充代理图标的框架。这里不需要急于绘制代理图标,因为还不知道实际有没有代理或存在哪些代理。再根据代理的矩形框图相对 panel1 位置绘制出相应的线条,用于连接 16 个测量仪器。

(4)类似步骤 3,区别是 for 循环 16 次,每次绘制正方形,用作填充测量仪器图标的框架。

1.3　拓扑图的生成与实现

前面已将三级拓扑层次结构框图绘制出来,接下来需要在其基础上绘制出实际局域网中设备的图标,这样就能确定局域网中有哪些设备,以及这些设备的连接关系。前面已提到 TreeView 控件的局限性,所以将它作为辅助的拓扑显示,而将文中设计的拓扑用于主要的拓扑显示,该主要拓扑依赖于辅助拓扑的拓扑关系及数据,而该辅助拓扑依赖于数据表 t_RunningTreeView。所以拓扑图的生成[6]步骤大致如下:

(1)生成 TreeView。首先从 t_RunningTreeView 中读取数据存入 DataTable 内存数据表中,利用 DataT-

able 类的对象的 Select 方法,即 DataRow[ ] dr = dt. Select(" pid = " +pid),通过 foreach 递归的从根节点,即 pid 为 0 的节点开始遍历 dr;每次遍历实例化新的 node 对象,用于在 TreeView 中添加新节点,即 TreeNode node = new TreeNode( ),将 dr 对象的每行值与 node 的相关属性绑定,再把 node 通过 Add 方法添加到 TreeView;在这个过程中还要进行总代理个数、总测量仪器个数等数据的统计。

(2)将 TreeView 中的信息保存到相应数组中,以便将 TreeView 辅助拓扑的拓扑关系及数据与主拓扑框图进行数据的绑定,即通过 foreach 语句递归地遍历 TreeView 中 nodes 节点,根据节点层次级别等属性判断,将 node 的信息存储到代理或测量仪器相关数组中,并做相关的数据统计,代码[7]如下:

```
private void tp_binding(TreeNodeCollection nodes)
{
foreach(TreeNode node in nodes)
{
if(node.Level==1)  //代理
{
agentName[i1]=node.Text;  //代理名
imgIndexAgent[i1]=node.ImageIndex;//代理图标索引
if(node.Nodes!=null)//该代理下还有孩子节点,即测量仪器
{
childNodeCount[k1]=node.Nodes.Count;//统计该代理下的仪器数
k1++;
sub_tp_binding(node.Nodes);//统计测量仪器相关数据
}
i1++;
m1++;
}
tp_binding(node.Nodes);//递归
}
}
private void sub_tp_binding(TreeNodeCollection nodes)
{
foreach(TreeNode childNode in nodes)//遍历代理下的测量仪器
if(childNode.Level==2)  //测量仪器
{
devName[j1]=childNode.Text;//存储测量仪器名
imgIndexDev[j1]=childNode.ImageIndex;//测量仪器图标索引
j1++;
n1++;
}
sub_tp_binding(childNode.Nodes);//可以注释掉,因为没有四级节点
}
}
```

(3)根据所点击的某一组,获取该组索引,利用索引、代理框图的坐标及步骤 2 得到的节点信息,通过 for 循环依次绘制出四个代理图标,并绘制出代理名等文字说明,对于不存在的代理用专门的特定图标替代,不需要绘制代理名等文字说明;代理绘制完成后,统计所查看组的前面所有组代理所连接的测量仪器总数,即:for(int i=0;i<4 * index;i++) j1 += childNodeCount[i];这样就知道了当前组中测量仪器图标的起始索引,然后结合该索引,通过双重循环(第一重循环为每个代理,第二重循环为每个代理下的四个测量仪器)和判断,依次绘制每个代理下所连接的测量仪器的图标,并绘制出代理名等文字说明,对不存在的测量仪器,用特定图标替代,不需要绘制代理名等文字说明。由于篇幅有限,不具体介绍代码,最后实现的拓扑是查看第一组拓扑显示的操作,见图 1(左侧为 TreeView 目录树图,右侧就是基于 TreeView 的拓扑图)。

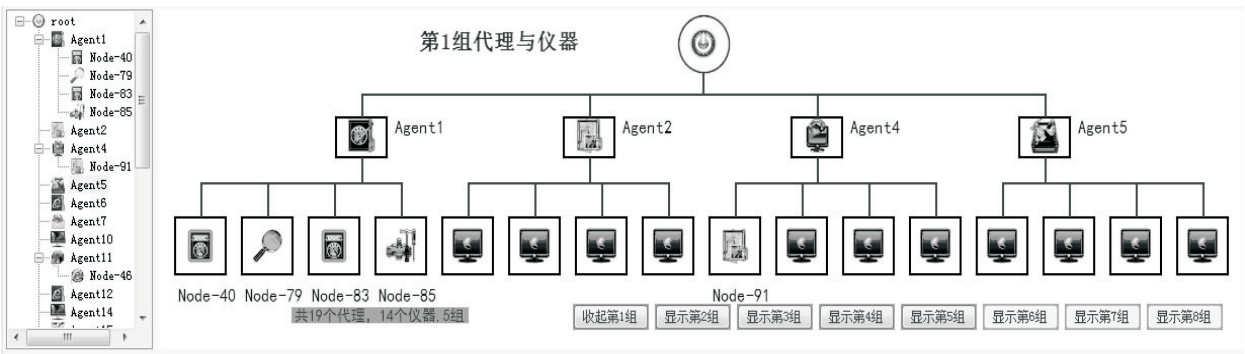

图 1 网络仪器拓扑

2 拓扑图的动态刷新

拓扑图的刷新实质上是清除已有的 TreeView 目录树状图和节点数据,再重新生成一次它们,基本原理还是根据 t_RunningTreeView 数据表中的数据生成 TreeView 目录树状图,再根据 TreeView 节点的数据绘制树状拓扑图。其缺点是无论拓扑图关系是否发生变化,就去刷新[8],人为控制时消耗人力,程序轮询控制

时,在时间间隔上又有局限性。要实现动态刷新,刷新原理不变,区别是当局域网的代理或测量仪器连接状态的改变导致拓扑图改变时,更新 t_RunningTreeView 数据表,再进行程序控制的刷新。

获取[9]局域网中拓扑图连接状态改变的方法有三种,三种方法相互协调,共同完成各自的任务,缺一不可。由于代码较多,限于篇幅,这里只概述流程,分别为:

(1)DHCP 自动发现[10]。这是后台模块调用的一个开源 DHCP,目的是发现局域网[11]中的代理开发板,并为其分配 IP 地址,然后将 IP 地址和代理的 MAC 地址推送给监管平台,推送指的是后台和监管模块之间采用进程间的 socket 通信实现;监管平台专门创建并启动了一个后台线程去接收该推送的任务,接收到推送信息后,将线程将字节数组中的数据反序列化成对应的结构体,并将得到的有效数据显示在监管平台主控界面的运行栏中,还要写入日志文件,并通过 MAC 在 t_AgentIDMAC 数据表中查找对应代理编号,再根据 IP 在 t_RunningTreeView 表中查找对应的 ID 号,如果没找到,说明该代理不存在,需要将该代理信息添加至数据表 t_RunningTreeView 中,表明拓扑图连接状态已改变。反序列化和结构体[12]代码分别如下:

```
//对 Socket 收到的字节数组数据进行反序列化
public Object BytesToStruct(Byte[] bytes,Type strcutType)
{
int size=Marshal.SizeOf(strcutType);//得到结构体的大小
//分配结构体大小的内存空间
IntPtr buffer=Marshal.AllocHGlobal(size);
//从 bytes 字节数组拷贝到内存空间
Marshal.Copy(bytes,0,buffer,size);
//将数据从非托管内存封送到结构类型的托管对象
object obj=Marshal.PtrToStructure(buffer,strcutType);
Marshal.FreeHGlobal(buffer);//释放非托管内存空间
return obj;          //返回对象
}

//反序列化后的数据结构为结构体,C#中结构的定义如下:
//引用 C#的 StructLayout 特性,说明结构的数据字段的物理布局
[StructLayout(LayoutKind.Sequential,CharSet=CharSet.Ansi,Pack=0)]
public struct DHCPInfo
{
//MarshalAs 特性表示如何在托管和非托管代码间封送数据
[MarshalAs(UnmanagedType.ByValTStr, SizeConst=20)]
public stringMACAddress;//MAC 地址字段
[MarshalAs(UnmanagedType.ByValTStr,SizeConst=20)]
public stringIPAddress;//IP 字段
}
```

(2)SNMP 协议的 Trap。代理 Trap 给监管平台的信息包括告警、事件和测量仪器的状态等信息,监管平台通过 Trap 的 OID 号进行字符串的解析和较为复杂的过滤、判断、映射及控制,获取测量仪器的插拔状态信息,更新 t_RunningTreeView。Trap 消息由代理模块发送给后台模块,后台模块进行一系列处理再通过 socket 套接字发送给监管模块。

(3)监管平台的扫描[13]代理线程。代理的上线有 DHCP 自动发现,测量仪器的插拔有代理的 SNMP 的 Trap。对于代理的断线,首先它无法通过网络 Trap,其次这里的 DHCP 不具备这样的功能,故只能依赖监管平台每隔一定时间去读取 t_RunningTreeView 表中所有代理的 ID 和 IP,然后通过 C#中 Ping 类和 PingReply 类来 Ping 每个 IP,即发送 ICMP 回送请求,并获取 ICMP 应答状态,如果应答状态不成功,则表明该 IP 对应的代理不存在,应该从数据库[14]中删除该代理的记录并删除该代理下连接的全部测量仪器,即更新[15]数据表 t_RunningTreeView,如果 Ping 通,ICMP 应答状态成功,则读取下一个 ID 和 IP,然后进行同样的操作。该扫描关键代码[16]如下:

```
Threadt_scan = new Thread(new ThreadStart(scanAgentIp));//创建线程
t_scan.Start();//启动线程
public voidscanAgentIp()
{
string sqlStr="select ip,id from t_RunningTreeView where pid=1";/* pid 为 1,表示代理,因为所有代理的父节点的 id 是 1 */
string connectionString="server=localhost;user id=root;password=lyb;database=xnmsdb";//数据库连接的必要信息
conn_s=new MySqlConnection(connectionString);//实例化数据库连接对象
conn_s.Open();   //打开数据库连接
MySqlCommand cmd=conn_s.CreateCommand();
cmd.CommandText=sqlStr;
MySqlDataReader read=cmd.ExecuteReader();
while(read.Read())//读取下一个记录行
{
string ip=read.GetString(0);  //获得该行第一个字段,即 ip
int id=read.GetInt32(1);   //得到 id
Ping pingSender=new Ping();//实例化 Ping 类
PingReply reply=pingSender.Send(ip,250); //发送 ICMP 回送消息,并得到应答
if(reply.Status!=IPStatus.Success)//判断 ICMP 应答状态,如果不成功
{
```

```
string sqlStr1 = " delete from t_RunningTreeView where ip=′" + ip + "′ or pid=′" + id + "′";//定义删除该代理及其所接的测量仪器的命令字符串
MyMysqlMeans. getsqlcom ( sqlStr1 );/ * 执行数据库操作,调用监管模块封装的数据库操作的静态类的相应静态方法 * /
load_tp = new Thread( new ThreadStart( createAutoFind) );
load_tp. Start( );
}
pingSender. Dispose( );//销毁对象
}
read. Dispose( );
Thread. Sleep(3000);  //隔 3 s 扫描一次
}
```

在上述三种方法中更新了 t_RunningTreeView 数据表后,就需要自动重新生成拓扑图,即调用绘制拓扑图所封装的方法[17],然而,这三种方式对应三个线程,这三个线程分别产生一个线程去专门执行重绘拓扑图的方法,如果再次产生的这三个线程都去调用该方法更新 UI 界面的拓扑图,会产生错误。C#中不允许这样的操作,所以要声明委托,在线程上执行指定的委托。关键代码如下:

```
Thread load _ tp = new Thread ( new ThreadStart ( createAutoFind) );
load_tp. Start( );
public delegate voidmy_autoFind( );//声明委托
public voidcreateAutoFind( )
{
this. Invoke( new my_autoFind( autoFindStart) );// 在线程上执行指定的委托
}
```

3 拓扑图的节点右击事件

节点操作事件是对拓扑图中的代理和测量仪器进行符合需求的各种操作,即在代理或测量仪器的图标区域通过单击鼠标右键形成各自的菜单栏,再根据菜单栏选定某一菜单项,完成该菜单项所提供的设备管理功能。panel1 面板鼠标单击事件的注册代码如下:

```
this. panel1. MouseClick+ =
newSystem. Windows. Forms. MouseEventHandler ( this. panel1_MouseClick);
```

对于实现 panel1_MouseClick(object sender, MouseEventArgs e) 方法的代码细节较多,这里只给出该方法的算法描述:

首先根据 e 对象判断单击的是哪个鼠标按钮,如果是左键,方法调用结束,否则对每一组拓扑中代理和测量仪器所占用的 20 个区域,逐个判断 e 对象的 X、Y 坐标是否在这些区域中,如果在(表明就不需要判断其他区域了,程序完成了整个功能后可以通过 break 跳出循环,并结束),计算出当前点击的是第 j 索引组拓扑图。然后再判断鼠标右击的第 i 索引区域编号是否小于 4,如果是,表明该区域是代理,否则该区域是测量仪器;其中 i 取值范围是-1<i<20,j 的取值范围为-1<j< 8。

如果是代理区域,即 i<4,需用 i<(agentNum-4 * j) 判断该区域是否存在代理,如果存在,通过 index = i+4 * j 计算出该代理在整个拓扑中的索引,最后根据该索引在该代理区域周围弹出菜单栏,即 agentMenu. Show(e. X+250, e. Y+150),程序结束;其中 agentNum 表示存在的代理总数。

如果是测量仪器区域,要用 index = 4 * j+(i-4)/4 计算出该区域是否属于 index 索引的代理下的区域,需用(index+1) < = agentNum 判断该区域附属的索引为 index 的代理区域是否存在代理。如果存在,通过 node = treeView1. Nodes[0]. Nodes[index] 获得 TreeView 控件中索引为 index 的代理节点,如果该代理节点下还有节点(测量仪器),通过 index1 = i% 4 计算出右击的测量仪器区域相对其附属代理下的 4 个区域中的索引号,再通过 index1<node. Nodes. Count 判断右击的测量仪器区域的测量仪器是否存在。如果存在,根据 treeView1. SelectedNode = node. Nodes[index1] 选定该测量仪器节点,在拓扑图的该测量仪器区域周围弹出菜单栏,即 devMenu. Show(e. X+250, e. Y+200)。

4 结束语

文中研究了局域网中网络仪器拓扑图的结构设计和自动发现、生成以及节点的右击事件,属于 LXI 网络仪器测控与管理系统项目中监管平台的一部分。该项目具备研究与实现的意义与价值,也具备创新性与实用性;该项目已被验收,运行正常,达到了预期目标。

参考文献:

[1] 李宥谋,薛 仙,黄 迪,等. 嵌入式 LXI 网络仪器的研究[J]. 计算机与数字工程,2016,44(9):1827-1831.

[2] 软件开发技术联盟. C#开发实战[M]. 北京:清华大学出版社,2013.

[3] 肖群健. 局域网拓扑发现技术研究与应用[D]. 广州:广东工业大学,2011.

[4] 张 玲,陈元春,孙 勇. 基于 GDI+的通用图形平台设计[J]. 计算机工程,2005,31(12):218-220.

[5] LIU Yu, WANG Minghui. A study on high speed drawing of graphics and image in GDI+ based application[J]. Advanced Display, 2006, 12(11): 66-69.

[6] 赵 玲. 网络拓扑发现算法的研究[D]. 长春:吉林大学,2011.

(下转第 158 页)

OpenCV 能够准确识别当前图像的时间,通过时间跳转功能可迅速定位至指定时间的图像。

图 3 实际应用效果

6 结束语

为了解决机载视频关键图像快速定位的问题,设计了基于 OpenCV 的试飞视频数据快速回放软件。采用基于 OpenCV 的图像识别技术对机载视频进行预处理,识别每帧处理后图像中的时间,然后在每帧视频数据前打上相应的时间戳,在进行回放时通过时间对比实现图像快速定位。该技术已经应用于多个型号的飞行试验视频数据回放中,可以通过指定时间快速跳转到该帧画面,正确性和可靠性得到验证,有效提高了视频数据回放效率,为保障型号试飞的高效顺利进行发挥重要作用。

参考文献:

[1] 张 杰,邹 强,晏 晖. 机载多路视频 PCM 遥测传输技术[J]. 计算机与数字工程,2013,41(5):805-807.

[2] 秦小文,温志芳,乔维维. 基于 OpenCV 的图像处理[J]. 电子测试,2011(7):39-41.

[3] 刘瑞祯,于仕琪. OpenCV 教程基础篇[M]. 北京:北京航空航天大学出版社,2007.

[4] 张汉灵. MATLAB 在数字图像处理技术方面的应用[M]. 北京:清华大学出版社,2008.

[5] 方 玫,喻擎苍,李华强. C++Builder 下基于 OpenCV 的数字图像处理[J]. 计算机工程与设计,2008,29(4):882-884.

[6] BRADSKI G, KAEHLER A. Learning OpenCV - computer vision with the OpenCV library[M]. [s. l.]: O'Reilly Media,2008.

[7] 孙凤杰,崔维新,张晋保,等. 远程数字视频监控与图像识别技术在电力系统中的应用[J]. 电网技术,2005,29(5):81-84.

[8] 沈庭芝,闫雪梅. 数字图像处理及模式识别[M]. 北京:北京理工大学出版社,2007.

[9] ANIL K J. Face detection in color images[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence,2002,24(5):696-706.

[10] PACKER J R. A system for fast erosion and dilation of bi-level images[J]. Journal of Scientific Computing,1990,5(3):187-198.

[11] CHENG Xinyu. Fast binary dilation/erosion algorithm using reference points[C]//Proceedings of the 2009 international conference on networking and digital society. [s. l.]: IEEE, 2009:87-90.

[12] NARAYANAN P J. Fast binary dilation/erosion algorithm using kernel subdivision[C]//Proceedings of Asian conference on computer vision. [s. l.]:[s. n.],2006:335-342.

[13] 胡页初,杨世明. 图像匹配算法研究[J]. 仪器仪表用户,2009,16(2):16-17.

[14] BAE J S, SONG T L. Image tracking algorithm using template matching and PSNF-m[J]. International Journal of Control, Automation and Systems,2008,6(3):413-423.

[15] 杨勇兵,何绪昊,戚其丰,等. 一种新型的快速模板匹配算法[J]. 电子工艺技术,2010,31(3):128-131.

[16] 党晓军,尹俊文. 一种基于模板匹配的运动目标跟踪方法[J]. 计算机工程与应用,2010,46(5):173-176.

(上接第 154 页)

[7] 周羽明. NET 平台下 Windows 程序设计[M]. 北京:电子工业出版社,2010.

[8] 周 静. 计算机网络拓扑自动发现及可视化的研究与实现[D]. 广州:华南理工大学,2010.

[9] 段东华. 基于 SNMP 协议的网络拓扑发现技术的研究与实现[D]. 济南:山东大学,2015.

[10] 薛 亮. 计算机网络多层拓扑结构的自动发现[J]. 科学技术与工程,2009,9(22):6682-6686.

[11] 静永文. 计算机网络自动拓扑发现技术的研究[D]. 保定:华北电力大学(河北),2003.

[12] TROELSEN A. Pro C# 5.0 and the. NET 4.5 framework[M]. [s. l.]: Apress,2012.

[13] LI Dancheng, ZHENG Chen, HAN Chunyan, et al. Research and application of multiple spanning tree network topology discovery algorithm[C]//Advances in computer, communication, control and automation. [s. l.]:[s. n.],2011:165-172.

[14] 秦 婧,石叶平. 精通 C#与. NET 4.0 数据库开发:基础、数据库核心技术、项目实战[M]. 北京:清华大学出版社,2011.

[15] 兰 洁. ADO. NET 数据库访问技术[J]. 电脑编程技巧与维护,2009(22):46-47.

[16] RICHTER J. CLR via C#[M]. 周 靖,译. 第 4 版. 北京:清华大学出版社,2015.

[17] WANG Zhangchao, ZHANG Yan, ZHANG Dedong, et al. An algorithm and implementation of network topology discovery based on SNMP[C]//Proceedings of 2016 first IEEE international conference on computer communication and the Internet. Wuhan, China: IEEE,2016.