上海交通大学大学生创新计划项目个人总结

——基于Android多屏幕拼接的蓝牙通信与触屏控制

上海交通大学 软件学院 王绍宇5080379072

Android蓝牙基本概念与使用方法

蓝牙，是一种支持设备短距离通信（一般10m内）的无线电技术。其数据速率为1Mbps，被广泛应用在移动设备的近距离通讯中。Android系统同样为程序员提供了蓝牙的接口，让手机之间的通讯成为可能。

我们蓝牙通信部分是参考Android SDK 中的Blue Tooth Chat（是一个基于蓝牙连接的聊天程序）的例子，完成我们对控制信息的发送和接受。

通常，蓝牙通信分为以下几个步骤：****

1. 搜索附近开启蓝牙功能的设备
2. 发起蓝牙连接请求，等待对方回应
3. 对方回应，建立连接
4. 开始通信

在第一次建立好连接之后，设备会自动存储配对好的远程设备，下次连接时可省去搜索和发现的过程，而直接进行连接即可。

需要注意的是，蓝牙连接的过程，也是一方（server）监听，等待连接，另一方主动（client）发送连接请求。这个通信的通道是通过用客户端与服务器之间共享的唯一UUID号来标识的。

下面阐述一下Android平台上蓝牙操作的要点。

首先，要操作蓝牙，先要在AndroidManifest.xml里加入权限

<uses-permissionandroid:name="android.permission.BLUETOOTH\_ADMIN" />

<uses-permissionandroid:name="android.permission.BLUETOOTH" />

会使用到的类：

android.bluetooth.BluetoothAdapter

android.bluetooth.BluetoothDevice

android.bluetooth.BluetoothServerSocket

android.bluetooth.BluetoothSocket

BluetoothAdapter蓝牙适配器，可以控制蓝牙功能的开启和关闭，获得蓝牙设备基本状态的类。

其中的方法getDefaultAdapter()是获取默认BluetoothAdapter的唯一方法。

接下来，如果client端知道远程设备的mac地址，那么可以用BluetoothAdapter类中的getRemoteDevice()方法，得到BluetoothDevice的对象。也可以通过BluetoothAdapter类中的getBindedDevice()方法得到所有已经配对的设备的集合，从中选择一个设备来连接。

这样，知道了待连接的device，通过BluetoothDevice的方法createRfcommSocketToServiceRecord()可创建一个client的socket，该方法参数为UUID，即可标识到底是哪个进程的通信通道。有了上面这个socket，即可用connect方法建立连接，用getInputStream和getOutputStream方法来用流的方式控制输入输出，通过蓝牙通信了。

而对于server端，在得到了BluetoothAdapter之后，可调用listenUsingRfcommWithServiceRecord()方法（需提供一个无关紧要的name和与客户端协商好的UUID），方法返回BluetoothServerSocket，该socket只在连接过程起作用，它的accept方法返回后，产生一个BluetoothSocket，此时，如果一对一通信，BluetoothServerSocket已经没有作用了，将其关闭释放即可，后面的通讯只使用BluetoothSocket即可完成。

蓝牙通信带宽有限，用于传输大量信息时可能会出现问题，所以我们还是用它来进行传输少量的控制信息。

触屏控制

使用触屏，是Android只能手机的一大特色。用触屏可以轻松便捷地实现许多人性化的功能，指头动一动，就可随心所欲地完成酷炫效果，给人一种很酷的感觉。

我们做的屏幕拼接图片的功能，也要达到一种人性化的水准，所以，触屏控制是需要细细研究的地方。

下面阐述一些触屏相关技术上的要点。

触屏动作的响应，是通过重载onTouchEvent回调函数完成的。当产生触屏动作时，该函数被自动调用。函数的参数将事件Event对象传入，其中包含了事件的类型以及相关数值。大体上event有三类：DOWN按下，UP抬起，MOVE移动。任何动作都是这些基本元素组合而成的。

一般处理触屏事件的代码框架格式如下：

public boolean onTouchEvent(MotionEvent event){

switch(event.getAction())

{

case MotionEvent.ACTION\_UP:

……

break;

case MotionEvent.ACTION\_DOWN:

……

break;

case MotionEvent.ACTION\_MOVE:

……

break;

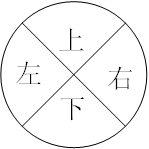
}

return true;

}

如拖动动作的识别。首先，收到DOWN事件，可忽略。因为后面将收到MOVE消息，而MOVE消息产生的前提就是已经按下了。MOVE的动作在UP消息到来之前，都是不被理睬的。因为，用户可能将手指按在屏幕上随便划，但只有最后一个动作，即离开屏幕瞬间手的移动方向才是要被执行的动作指令。所以，我们判断划动方向是在收到UP事件时，检查之前最后MOVE的方向。

我们大致将移动分为上下左右四个方向。方向的分区是按照四十五度对角线作为区分的。

而且，为了避免噪音动作的影响，我们会忽略位移非常小的移动（近似点动）

对于放大和缩小操作，我们利用Android的两点触控。当两个手指同时按下并拖动，并且距离增大了，我们即可判断是放大动作，反之，是缩小动作。

我们可以利用event.getX()和event.getHistoricalX()进行比较，得到手指在X方向的位移，同理得到Y方向的位移。而四十五度的判断，就是比较X方向的位移与Y方向 的位移的绝对值大小。

至于多点触控操作，相应的getX, getY, getHistoricalX, getHistoricalY等函数均有另一个版本，多接受index参数，这个参数指明是哪个点的坐标。而现在的Android触屏最多只支持两点操作。所以index范围就是0和1。

Gallery与图片索引

由于Android提供查找图片的媒体库的内置的Activity，所以我们在查找手机中所有图片的时候，可以直接使用一个Intent，跳到这个Activity中，用户选择了图片后，可以将图片加载为资源，返回它的uri给我们，以供调用。

打开Gallery选择图片的代码如下。

**public** **void** openGallery(View v){

Intent intent = **new** Intent();

intent.setType("image/\*");

intent.setAction(Intent.*ACTION\_GET\_CONTENT*);

startActivityForResult(intent, *OPEN\_GALLERY*);

}

上面的代码执行后，将跳到系统提供的Gallery的Activity，这个Activity是有结果的，所以我们用startActivityForResult函数，预示着后面将处理Activity返回的结果。

而结果处理函数是一个回调函数，如下：

**public** **void** onActivityResult(**int** requestCode, **int** resultCode, Intent data) {

**switch** (requestCode) {

**case** *OPEN\_GALLERY*:

**if** (resultCode == Activity.*RESULT\_OK*){

setContentView(R.layout.*start*);

Intent intent = **new** Intent();

intent.setData(data.getData());

intent.setClass(**this**, Splice.**class**);

startActivity(intent);

}

**break**;

}

}

可是，由于要使用媒体库的内置的Activity，提供的API，是将用户选定了的图片加载入资源，直接将该资源以URI方式返回给开发者。而这个URI并不包含文件的文件名信息，而是采用序号标识资源的。

这样就带来一个问题。这张图片究竟如何识别它，怎样告诉另一台手机，应该显示哪张图片。

既然拿不到文件的元数据等，那我们就直接基于内容本身来标识图片。所以，使用Hash方法将图片的数据进行散列，作为图片文件的唯一标识，在发送消息时，可以指定这张图片的Hash，发送这个Hash到对方手机，来寻找相同内容的图片。这样做的好处是可以直接根据图片的具体内容索引特定的图片，标识它们的唯一性，而且又不需要将文件整体这么大的数据量在网络中传输，极大地节约了资源。

图片查看操作——缩放、移动、下一张

为了做到媒体库中图片的查看功能，我们决定加入图片的缩放，移动和上一张下一张等功能。用户操作都是采用触屏操作，可参见前面触屏部分介绍的各种操作。

对于图片的缩放移动，对于拼接的屏幕来说，与单机对图片的操作略有不同。

在默认情况下，我们对图片进行横向拼接。初始时，图片被从中间的一条竖线分为左右相等的两部分。这时，我们的实现方式是左边手机显示原图片的左半部分，又边手机显示原图片的右半部分。当某一手机发送了放大操作指令出来，那么要讲这个指令和当前的放大倍数传递给对方手机，此时各手机应重新计算需要显示图片的部分。

当某手机触屏收到移动图片指令时，也应将移动指令和显示的位置信息通过蓝牙发送到对方手机。并且两边更新显示。

具体的图片移动和放大后，屏幕上显示位置的计算公式如下：

处于左端位置的手机需要截取屏幕的( x, y, width, height ) ：

( posx, posy, (w/2+posx)/zoomLevel, (h+posy)/zoomLevel )

处于左端位置的手机需要截取屏幕的( x, y, width, height ) ：

( ( posx+w/2)/zoomLevel, posy, (w+posx)/zoomLevel, (h+posy)/zoomLevel )

其中posx，posy，zoomLevel分别是需要传输的移动图片的x，y位置，和放大率。