[C#实现按键精灵的'找图' '找色' '找字'的功能](https://www.cnblogs.com/JimmyBright/p/4355862.html)

原文地址：

[*http://www.cnblogs.com/JimmyBright/p/4355862.html*](http://www.cnblogs.com/JimmyBright/p/4355862.html)

[**背景：**游戏辅助功能通常使用按键精灵编写脚本，按键精灵的最大卖点就是能够找到画面中字，图，色，这对于模拟用户鼠标操作至关重要，这能找到道具，找到血量，实现自动打怪，自动补血，自动买卖道具，博主闲来无聊，看到一款按键精灵实现的辅助，于是乎想用WPF也写一款辅助工具，实现其核心的找图找色等功能。博主测试，对于背景复杂多变的画面，找不变图的成功率达到100%，找带透明的图，比如文字，能达到90%以上。默认您已经知道一个颜色值由argb构成，每个值范围都是0~255。网上发现不少人询问过该问题，几乎没有比较全面的解答，今天本博主自己写了这些功能的代码，C#同学们，以后可以用这几个函数实现你的游戏辅助了哦(⊙o⊙)](http://www.cnblogs.com/JimmyBright/p/4355862.html)

**找色：**

****

方法中的容错范围，默认设置为10。R、G、B三者的范围都是0~255，容错为10，就表示每个范围都可以在10上下波动，下面还会有容错的概念

**找图：**



在一张大图中截取一个矩形小图，然后在任意包含该小图的图片中找到该小图的坐标位置

**找字：**

找字比较困难了呢，因为文字是一种镂空的图像，不像上述找的是非镂空图像，代码：

定义结构体：

struct NumBody

{

public int num;//数字

public int matchNum;//匹配的个数

public int matchSum;

public double matchRate;//匹配度

public System.Drawing.Point point;

public List<System.Drawing.Point> bodyCollectionPoint;//该数字所有像素在大图中的坐标

}



特别注意：有了这个方法还是不能找到你要的文字的。要先处理文字，下面举例：

例如在这张图片上找到朋友的朋字的坐标位置：



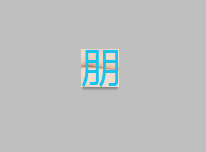
1：打开你的PS，先将图片放大，看到像素方块为止，然后将朋字的范围圈选住，注意稍微比字圈选的大一点，像这样：



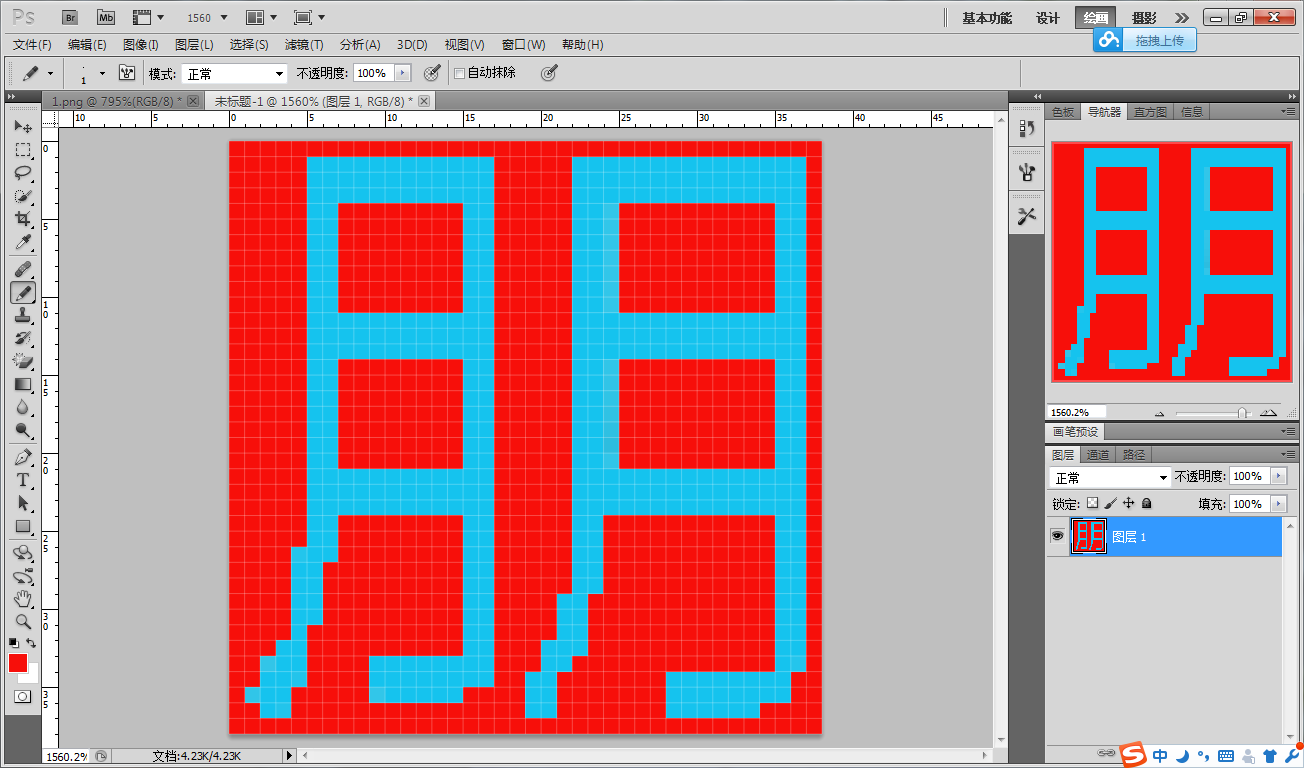
2：按住CTRL+C，然后CTRL+N，出现对话框：(教教大家使用PS^\_^)



3：将背景内容选择透明，按确定，再按CTRL+V复制图像



4：将这个图片放大到看到像素为止，将所有非字体的位置全部用铅笔工具涂上同一种颜色，



5：涂完了之后将这张图片保存下来，这张图片就是我们要查找的“朋”字，图片是这样的

https://images0.cnblogs.com/blog2015/274393/201503/211854319375167.png

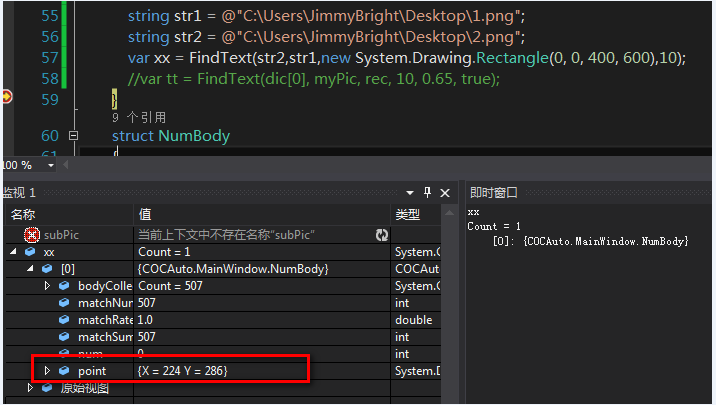
**6：我们需要的就是第五步的图片和第一张底图，下面见证奇迹的时刻到了。**

string str1 = @"C:\Users\JimmyBright\Desktop\1.png";

string str2 = @"C:\Users\JimmyBright\Desktop\2.png";

var xx = FindText(str2,str1,new System.Drawing.Rectangle(0, 0, 400, 600),10);

**str1是我们的底图，str2是第五步的那张处理后的文字图片，xx就是我们最后需要的文字的位置坐标，我们运行看看。下面截图运行结果：**

****

**显然最后我们查找的文字在图片中的坐标为(224,286)，大家可以下载那张图片验证**

**找数：**



　　你以为找到文字就算完了吗？No，找数字才是最困难的，为什么呢？有人会问，数字难道不也是文字吗，不也可以通过PS处理数字达到查找其位置的目的吗？对的，数字也是文字，我们将需要查找的数字0~9全部PS处理，就能查到它们的位置了。但是有一个问题啊，游戏中用数字表示的地方通常是一连串的数字，这些数字里面包含0~9的任意组合。所以我们需要这样处理：

　　我们从0~9依次查找指定区域，记录每次查找的结果，没查到的数字不必记录，对查到结果的数字再按照X坐标排序，因为在X坐标越小，数字越靠左边。

还有一个严重的问题，例如38,14，这样的数字会很讨厌，为什么呢，我们会再8当中查找3，在4当中查找到1，这会对我们的数字识别产生重大误差，所以下面我也写了一个方法对这个问题做了处理，代码：

**其他方法：**



结束语：以上代码本人实现了找颜色，找图片，找文字，找数字的所有功能，希望对朋友们能有所帮助。