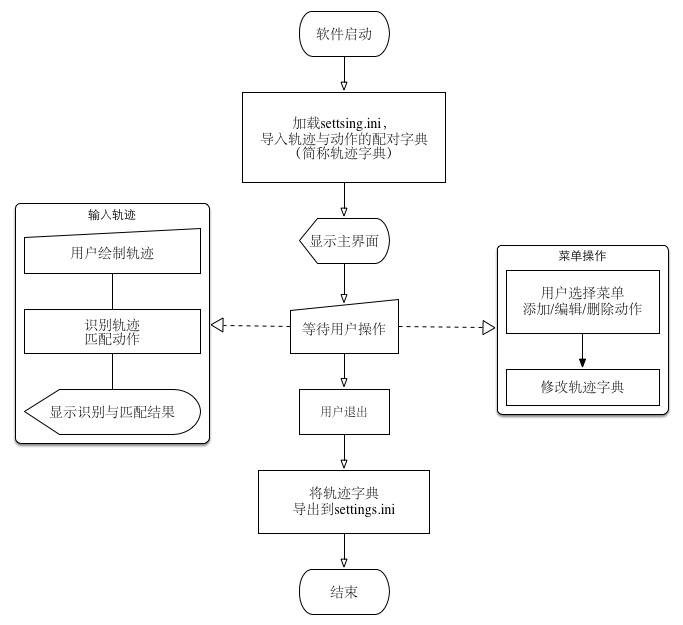
大作业报告

宋艺博 2011013268 王学成 2011013252 庄晨帆 2011013246

# 选题

我们小组用汇编语言实现鼠标手势识别程序Quick Launcher。 通过识别鼠标轨迹，快速启动程序，并且支持用户来新增、删除、编辑手势以及对应的动作。

# Quick Launcher使用



文字简介…TO DO

# 代码结构

我们采用了类似MVC的架构方式，分为：

* 前台主界面（View）：监听用户操作，例如输入轨迹、请求编辑手势等
* 中间传递模块：负责对前台数据进行初步处理，并传递给后台进行处理，再将后台返回的数据进行包装，传送给前台用于显示
* 后台数据处理：对得到的数据进行识别操作，与“字典”中的数据进行匹配；对字典进行增、删、查、改操作。

插入图片 TO DO

这种分层的架构方式，有诸多优点：

1. 让代码结构变得清晰、有逻辑
2. 便于多人合作，每位组员负责一个模块，各有专攻，不需要对整个工程的代码都了解，只需要与组员商议好接口即可。使用git工具来管理代码，冲突少。
3. 便于测试。各位模块负责人确保自己模块的正确性。在后期测试时，只要关注各个模块的输出是否正常。如果有异常发生，交给相应的组员，debug的效率大大提高。
4. 便于迭代开发。每次开发重点“升级”一个模块，保持接口统一，其他模块基本不需要变动，省去了大量的修复工作。并且借助其他模块来对当前的修改部分进行测试，省时省力。

## 自定义结构体ACTION

在declaration.inc中定义了结构体ACTION：

ACTION STRUCT

len dd 0 ; length of seq

seq dd 32 DUP(0) ; direction seq

path db 1024 DUP(0) ; path of responsive action/command

tip db 1024 DUP(0) ; tip of responsive action/command

pathType dd 0 ; action/command type

ACTION ENDS

## 导入/出配置

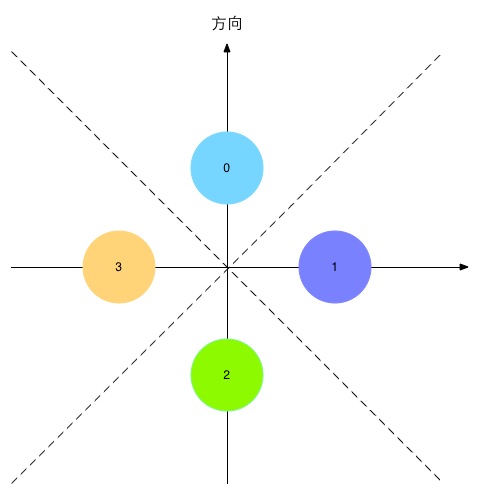
文件settings.ini用于保存用户的手势字典，在quick launcher启动时自动载入到内存中，settings.ini的每一条记录（一行）对应一个ACTION结构体，格式为:

#type #len #seq[1]#seq[2]…#seq[len] path|tip

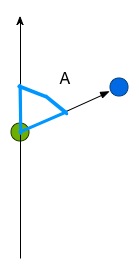
例如：【TO-DO】

## 识别方向

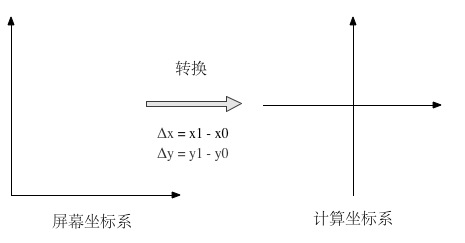
方向定义如下：



计算从点P0（上一个绘制点）到P1（当前绘制点）的移动方向，利用向量与y轴的夹角，得到当前的移动方向（根据上图中的角度进行判断，）：



需要注意的是，屏幕坐标系与我们常见的坐标系（称为计算坐标系）略有不同，需要进行一个简单的转换：



具体代码见Recognise.asm中recognise函数。

## 匹配手势

考虑到在实际情况中，用户不会记住大量的手势，因此手势字典(actionMap)预设可容纳1024个手势。在这个数量级上，逐一枚举手势进行匹配就能达到令人满意的效果。

为了更好的用户体验，我们采取了前缀记录与精确匹配的方法:

首先，为手势字典中的每一个ACTION维护一个标志变量，用prefixMatchArray数组记录第k个手势是否能匹配上，即：

prefixMatchArray[k] = 0 actionMap[k]的前缀能匹配

prefixMatchArray[k] = 1 actionMap[k]的前缀不能匹配

每当后台监听到一个有效点（@wxc，你那里先定义一下？有效点的定义前前文），不妨设为是当前输入手势的第m个有效点，就与当前prefixMatchArray[k] = 0的actionMap[k]的第m个方向进行匹配，若失败了，则 prefixMatchArray[k] = 1。

因此，无论鼠标轨迹有多长，总的匹配次数不会超过actionMap中所有ACTION.len之和，是一个程序可以接受的常数。

用一个全局变量bestMatch记录是否有匹配手势，若无，则置为-1，由负责执行命令的函数来监听bestMatch，触发相应的操作。

# 迭代历程

1. 首次开发
   1. 确定选题，讨论方案的可行性与功能的实用性，找到软件的定位，明确各位组员分工
   2. 搭建整体框架，创建一个最简单的demo，商定模块之间的接口，统一数据存储方式
2. 第二次开发
   1. 采用简单粗暴的方式对轨迹进行识别，算法虽然粗糙，但是准确率不低
   2. 完成轨迹的绘制
   3. 实现启动其他程序的功能
3. 第三次开发
   1. 对识别算法进行改进
   2. 采用“双缓冲”，解决主界面更新时“闪烁”的问题
   3. 实现打开虚拟文件（例如我的电脑、回收站）等等的功能
   4. 完成菜单栏、弹框等界面
4. 第四次开发
   1. 实现导入、导出程序配置
   2. 添加实时匹配并显示识别方向的功能
   3. 实现弹框响应，支持用户新加手势与编辑动作路径
5. 第五次开发
   1. 完善手势的添加与编辑功能
   2. TO DO

# 亮点

1. 增删查改什么的
2. 选择是否开机启动
3. 全局热键，CTRL + Q可显示/隐藏Quick Launcher，并且显示在所有窗口的最前
4. 再想想

# 心得总结

“实践是检验真理的唯一标准”……@zcf待我写