Multitouch Technologies Handbook
Introduction for Multitouch Setup and Applications
Published by Natural User Interface Group
Translated by mt2a.com

多点触摸技术手册

Multitouch Technologies Handbook

简体中文版

NUI Group & mt2a.com

Multi-Touch Technologies

NUI Group Authors

1st edition [Community Release]: May 2009

Cover design: Justin Riggio

Contact: the-book@nuigroup.com

Chinese Edition © www.mt2a.com

VERSION 1.00 BUILT ON 20090714 CHS/CHT

绪论

关于本书

自然用户界面小组（Natural User Interface Group）创建于 2006 年，以互动媒体探索以及开源机器遥感技术为中心，开发受益于艺术、商业、教育等相关的应用，是全球规模最大的关于自然用户界面的开源社区。

NUI Group 为在人机交互领域上感兴趣的开发者提供一个彼此相互交流的环境，涉及到的课题有：拓展现实(AR)、声音或者手写动作识别、触摸计算、计算机视觉图形以及数据可视化等。

这本书是对于在维基上各章节的汇总，在多点触摸领域上是唯一的一本书籍。在此，我们十分感谢所有推动 NUI 多点触摸社区发展的爱好者，贡献比较大的成员会在下面列出来。

我们相信这本书将会让大家带着欢悦的心情去阅读，也希望读者们积极地通过 the-book@nuigroup.com(中文版可登陆 www.mt2a.com 进行讨论)给予我们评论、建议，我们的大门永远地为大家敞开着，希望有更多的爱好者加入我们的行列中。

关于作者

Alex Teiche 在人机交互领域上研究多年，特别是在合作式多点触摸交互里有着浓厚兴趣的热衷者。他利用 LLP 多点触摸技术完成了自己的多点触摸桌子，用了数个月的时间参与 PyMT 的开发工作。他现在正在为爱好者们尝试如何创造一个薄型且简易的多点触摸设备。

Ashish Kumar Rai 印度贝拿勒斯印度教大学电子工程技术学院的本科生，他的主要贡献在于发展 QMTSim 模拟器，目前他的工作是推动以及优化 NUI 的硬件与软件方面的解决方案。

Chris Yanc 专业的互动设计师，在俄亥俄州克里夫兰的一家网络营销机构工作超过四年时间。在他搭建自己的 FTIR 多点触摸桌子后，我全心利用 Flash 和 Actionscript 3.0 结合 CCV 和 Touchlib 来开发多点触摸的应用。他一直在 NUI 社区里面分享着自己的进展以及经验。

Christian Moore 开源技术的先锋以及是 NUI 的灵魂人物，作为 NUI 的创始人之一，他带领着社区发展着艺术形态人类感应技术，同时他致力于推动开源的标准，统一社区的各种开发。

Donovan Sloms 南非高登省比勒陀利亚大学多媒体就读，在 2007 年第一季度搭建自己多点触摸桌子后，致力于利用.NET 和 ActionScript 3.0 开发多点触摸的应用，目前他在南非能够提供多点触摸设备商业应用以及多点触摸应用软件的解决方案。

Görkem Çetin 以开源界的灵魂人物来定义自己，博士研究焦点在开源软件上的人机交互开发。在技术领域上，他已经出版了五本书，以及在各种会议上和

杂志上发表了许多相关的文章。

Justin Riggio 专业的互动开发师和设计师，目前为止已经有五年的时间担任自由职业者了，他工作和居住在纽约地区，有时候也能够在新泽西州海岸冲浪时看到他。他已经搭建了 DI 的多点触摸桌子，同时也在测试利用 LCD 为显示屏的 LLP 多点触摸设备，你可以通过阅读器看到他最新关于 DSI 多点触摸设备的草图以及一些 AS3.0 多点触摸文件。

Nolan Ramseyer 美国加州大学圣地亚哥分校生物技术工程（以生物技术为背景的计算机和多媒体学科）的学生。利用空余的时间来搭建和研究基于光学多点触摸技术，通过他的经验和知识帮助了许多爱好者实现自己的项目和目标。

Paul D'Intino 澳大利亚墨尔本 Pioneer Electronics 公司的技术培训经理，在 2003 年，他取得了娱乐和服务电子技术三级认证，走上了修理消费性电子设备的道路，主要专注在等离子电视上。因为他对电子科学热爱，现在参与多点触摸设备与应用的发展，同时也协助 NUI 的开发。

Laurence Muller (M.Sc.) 荷兰阿姆斯特丹大学科学可视化和虚拟现实研究组的程序开发员，2008 年他完成了以“多点触摸显示：设计、应用、绩效评估”为题目的论文，目前他正在开发多点触摸的相关软件。

Ramsin Khoshabeh 毕业于美国加州大学圣地亚哥分校电子与计算机工程系博士学位，现在分布式认知与人机交互实验室的科学认知部门工作，他的研究重点在于开发可交互的工具以及大型数据集视频的分析。在他工作的范围中，他开发了 Multi-Mouse，用于多点触摸设备上 windows 平台的鼠标操作，也开发了 Video-Navigator，用于多平台的视频引导软件。

Rishi Bedi 美国马里兰州巴尔的摩的一名学生，在用户交互设计上非常感

兴趣，懂得部分语言的编程以及硬件的开发。作为一个 NUI 社区上的积极成员，他希望能够搭建自己的多点触摸装置，为爱好者们提供帮助，他同时也喜欢平面设计，他设计了这本书的封面，曾在报刊和电子媒体公司工作过。

Mohammad Taha Bintahir 加拿大多伦多安大略技术学院电子工程学的本科生，他专注于微电子学和编程学。目前，他正在做自己项目课题的研究，包括了电子和基于光学的多点触摸装置。

Tomas Hansen 爱荷华大学的博士生以及是开源领域的支持者，他感兴趣的研究包括信息可视化、计算机图形，重点关注在类似多点触摸显示新界面的人机交互领域上。他的工作得到了爱荷华大学数学与物理科学基金项目的支持。

Tim Roth 在瑞士苏黎世设计学院学习交互设计，在学习期间他专注与交互设计、粒子信息可视化以及人机交互的新发展。他研究生项目是帮助一家私人银行解决客户与顾问的问题，通过多点触摸显示设备方式来帮助银行顾问解释以及可视化一些复杂的商业计划给客户。

Seth Sandler 毕业于美国加州大学圣地亚哥分校跨学科计算机与艺术的本科学位，学习的重点在与音乐。在学习的最后一年里面，他开始研究和搜索多点触摸相关的资料并开发了命名为“AudioTouch”的多点触摸音乐应用项目。他是 NUI 社区上受人注目的成员，参与介绍“MTmini”和 CCV 的开发。

编辑者

这本书是由 Görkem Çetin, Rishi Bedi and Seth Sandler 设计与编辑的。

关于译者

mt2a：张海云，交互设计从业者，从事未来概念产品设计多年，mt2a.com 创办人，致力于推动多点触摸技术的发展和应用。

联系方式：seeacloud@gmail.com

Kevin：刘得志（Kevin Lau），mt2a.com 硬件区版主。2006 年获得 Alias 公司 Maya 动画师认证；2007 年创办万臣品牌视觉设计团队；2008 年就读于德国不莱梅大学及德国不莱梅艺术大学数字媒体系，并参与数字媒体互动方向的研究；2009 年获得广东工业大学数字媒体工程系学士学位，热爱交互设计及用户体验探索，致力于推动中国数字媒体发展与应用。在 2009 年 mt2a.Com 组织的国内第一本多点触摸书籍《多点触摸技术手册中文版》一书中担任 MultiTouch Technology 部分的翻译。

联系方式：Lau.dezhi@gmail.com

Thons：张振邦(Thons Chang)，一位 90 后，高中生，mt2a.Com 软件区斑竹。自幼爱好并自学计算机，在小学时为本地的商场做过产品宣传片，并接触编程。初中时当过小闪客，但最后还是从 Flash 设计走上了编程，2007 年接触多点触摸技术，2008 年年底对多点触摸技术产生浓厚兴趣，并于 2009 年 mt2a.Com 组织的国内第一本多点触摸书籍《多点触摸技术手册中文版》一书中担任 Software & Applications 部分的翻译。

联系方式：thons@ymail.com

mt2a.com 是专业的多点触摸技术论坛，取 multi touch to all 之意，缩写为 mt2a，我们的愿景是让每个人都能体验到多点触摸。目前论坛属于初创，以学习和普及为主，将来会成立专门的开发团队，致力于多点触摸解决方案的开发。

热忱的欢迎每个喜爱多点触摸，喜爱交互设计的朋友都到论坛来，希望更多的人和我们一起，推动多点触摸的发展，我们永远欢迎你。

目录

第一章

多点触摸技术

本章预览

1.1 硬件导论

多点触摸指的是允许计算机用户同时通过多个手指来控制图形应用的一种表达方式，而多点触摸设备是由可触摸示设备（例如：计算机显示器、桌子、墙壁）或者触摸板组成，通过软件识别同时发生触摸行为的点。这与市场上常见的触摸显示屏（例如：计算机触摸板、银行的 ATM 柜员机）不同，市场上常见的触摸显示屏只能够识别单或者双点。

自然用户界面小组（NUI）和多点触摸硬件设计以及多点触摸手势设计，特别是在搭建一个真正的多点触摸硬件设备（例如：支持多于两个点的输入）上带来了关键性的变革。NUI Group 的目标在于提供一个开放式的平台，在这个平台上大家可以自由的交流多点触摸硬件和软件的知识，推动着整个多点触摸技术的快速发展。

作为硬件上的先锋，NUI Group 希望能够为在搭建低成本、高分辨率、开源式的多点触摸设备感兴趣的人提供一个多点触摸技术的信息资源中心。通过这个平台不断发展和壮大，多点触摸技术带来了许多惊人的开创，这不仅仅局限在多点触摸设备，还引发了更多相关的设备出现。到目前为止，已经有五项可以帮助爱好者搭建稳定的多点触摸平台的技术出现，它们分别是：由 Jeff Han 教授开创的受抑全内反射多点触摸技术（FTIR）；微软 Surface 桌子的背面散射光多点触摸技术（Rear-DI）；由 Alex Popovich 提出的激光平面多点触摸技术（LLP）；由 Nima Motamedi 提出发光二极管平面多点触摸技术（LED-LP）；由 Tim Roth 提出的散射光平面多点触摸技术（DSI）。

这五项技术主要是基于计算机视觉和光学为主的，除了这五项在 NUI Group 里占绝大多数之外还有一些其它的技术同样可以搭建多点触摸设备，它们包括声

波器、电容、电阻、动作捕捉器、定位器、压力感应条等。通常情况下，这各种感应器结合起来，就可以搭建一个特别的多点触摸设备。在本章节中，我们将会和大家探索这些提及到的五项多点触摸技术。

1.2 基于光学的多点触摸技术导论

基于光学（例如：摄像头）的多点触摸技术搭建起来的设备体积比例相对要大，但它的可拓展性、低成本以及容易搭建是其受到欢迎的原因。受抑全内反射多点触摸技术（FTIR)、正面和背面散射光多点触摸技术（Front and Rear DI)、激光平面多点触摸技术（LLP)、发光二极管平面多点触摸技术（LED-LP)、散射光平面多点触摸技术（DSI)，这些都是基于光学多点触摸技术的例子。

上述的每个基于光学的多点触摸技术都包含着光学感应器(通常为摄像头)、红外光源以及通过投影仪或者显示器为显示的屏幕，因为有这三个相同点，所以在系统的学习各项技术前，需要对这三部分有一定的认识和了解。

1.2.1 红外光源

红外线（英文简称：IR）是光谱上的一部分，刚好越界于人类眼睛可以看到的可见光，其范围长于可见光但短于微波，“近红外”（英文简称：NIR）处于红外光谱上的最低处，一般被认可的波长为 700nm（纳米）到 1000nm 之间。大多数数码摄像头的传感器对红外很敏感，所以我们通常看到的摄像头都加装一块可以滤除红外线的镜片，以便于摄像头只捕捉可见光，但这于我们所需要的相反，因此我们需要将一块可以滤除其它波长光只接收相对应红外线波长光的镜片替换原先的就可以达到我们所需要的目的。

在多点触摸技术中，红外光源主要作用于区别触摸表面的可视界面和物体或者手指痕迹。鉴于很多系统都以投影仪或者显示器作为显示的设备，因此如何让摄像头仅读取物体或者手指反馈的信息点是我们需要关注的，正如上面所说的，通过改装摄像头然后仅让其读取我们在触摸表面上所需要反馈的信息点即可。

大多数压克力生厂商在生产压克力时已经加强了可以削弱 900nm 以上的红

外线能力，这样可以帮助在作为窗户使用时是减少太阳的加热。很多摄像头在这方面上也做了修正，减少对 940nm 的敏感以及对减低太阳光的干扰。

红外发光二极管并不是我们真正需要的，但它的红外光是我们要的。在多数基于光学多点触摸技术中（特别是 LED-LP 以及 FTIR），红外发光二极管被运用的原因在于它们具有高效性地提供红外光源的优点。例如散射光多点触摸技术（DI）却不一定需要红外发光二极管，但这不排除具有红外发光二极管的红外光源组，而激光平面多点触摸技术（LLP）利用红外激光器作为红外光源。

通常情况下，发光二极管能够以“单红外发光二极管”或者“红外发光二极管带”购买：

单红外发光二极管：单红外发光二极管价格相对便宜而且可以很容易为利用 FTIR、DSI 以及 LED-LP 作为多点触摸技术的设备制作发光二极管框，而需要我们具备的知识是如何去焊接我们所需要的电路。目前最常用的型号是欧司朗的 SFH 485P，如果你想利用 LCD 作为显示屏的话，那么你则需要亮度更大的红外发光二极管。

红外发光二极管带：这是用柔性扁平电缆结合红外发光贴片结合起来的条带，十分便利，购买的时候已经是组装好的，只需要我们贴在压克力四边则可（公认质量最好的是美国的 environmentallights.com 提供的红外发光二极管带）。

红外发射器：这是用于 Rear DI 或者 Front DI 装置中的，这种方式的装置比其它的都要简易，只需要我们通过红外发射器将箱子内部照亮则可，但需要我们注意的是如何消除因为红外发射器引起的区域过亮问题。

在购买红外发光二极管前，需要十分注意的是看清楚发光二极管的参数表，波长、角度、功率，这些都是整个技术的重点。

波长：780-940nm，红外发光二极管在这个范围内最容易被大多数摄像头读取。波长越低，敏感度就越高等同于更容易分析压感。

功率：最低为 80mw。

适用在 FTIR 的角度：角度低于正负 48 度的不能够产生全内反射，而角度高于正负 48 度的则会产生红外线溢出压克力。为了确保其范围，可以利用高于正负 48 度的，但高于正负 60 度则会造成浪费（60-48=+/-12 度），这样子就会溢出压克力了。

适用在 DI 的角度：通常来讲角度越广越好 ，更广的角度产生的效果会更容易。

对于 DI 装置里面，很多人会有遇到区域过亮的问题。为了解决这个问题，我们建议将发光器反转照射，避免对着显示区域，同时我们需要为摄像头加上过滤片，简易的软盘过滤片可以产生效果，但效果并不好，我们建议用专业的过滤镜片来解决问题。

1.2.2 红外摄像头

简单的网络摄像头可以用于多点触摸设备上，但是我们需要对其进行改装。一般的网络摄像头或者摄像头将红外光过滤了，只让其读取可见光，所以我们需要做的是相反的工作，我们需要对其进行改装成只需要红外光可以被读取。通常情况下，我们只需要打开摄像头的盖子，然后将过滤红外光线的镜子去掉，换上可以过滤可见光的镜片即可，但有些可以很容易去除，而有些价格比较贵的摄像头会将这个具有过滤红外线功能的镜子整合在摄像头里面，我们无法去除。

不排除有些摄像头可以直接读取红外光线，但进行改装的摄像头运用起来的效果会更加好。

多点触摸设备的性能好坏取决于其运用的部件，因此你需要十分谨慎地去选择你所需要的部件。在购买摄像头之前，你需要明确的是自己的目的是什么？如果你仅需要搭建一个小的用来测试的多点触摸设备，那么一个简单的摄像头就足够了，但相反如果你需要搭建一个用来演示的多点触摸设备，你则需要考虑购买更好的摄像头了。

分辨率：摄像头的分辨率十分重要，分辨率越高的摄像头在读取物体或者手指的时候能够更加的清晰和容易，这对于需要做到十分精确的设备来说十分重要。对于一个小的多点触摸设备来说一个低分辨率的网络摄像头（320*240 像素）就可以胜任了，而大的设备则需要一个分辨率高的摄像头（640*480 像素）来达到其精确度。

帧率：帧率是指摄像头在一秒中内读取到的帧的数目，更多的快照意味着在定义的时间内我们具有更多的数据。为了让设备能够更好的读取我们移动产生信息点以及反应更加灵敏，我们至少需要 30 帧数每秒（FPS）。更高的帧数可以提供更好的流畅度和更好的用户体验。

接口：通常情况下，我们可以通过两个类型的接口来将摄像头和电脑连接。根据项目的不同，可以选择常用的 USB 接口或者专业的 IEEE1394 接口（常说的“火口”）。IEEE1394 摄像头因为对读取信息的衰减少而能够更好地将信息传送给计算机处理，一个对信息衰减越少的摄像头能够给设备提供更好的效率。

镜头类型：大多数网络摄像头都具有阻挡红外线的绿镜片，也具有避免图形变形的。但对于我们的目标，我们需要捕捉以及利用红外线。很多网络摄像头可以很容易去除滤除红外的镜片，这个镜片被放置在镜片的后面，具有红色反光的特性。但当有些摄像头无法拆除红外滤镜的时候，就需要将整个镜头进行更换。

网络摄像头一般都会用到 M12 型号的底座，工业摄像头系列（IEEE1394）通常需要另外购买镜头，另外不同型号的摄像头，也有不同的底座，例如：M12、C 或者 CS。

要选择一个好的摄像头不是一件容易的事情，幸运的是很多摄像头生产商会提供一个在线的镜头计算工具，这个工具通过输入两个参数便能够帮我们测量出在镜头与物体以及显示区域的长度和宽度的焦点长度，但要注意的是要注意选择好相对的型号。一个焦点长度比较低的镜头往往会产生很多不好的效果（例如：图像变形），这样子会干扰我们定位信息点，使得工作难以进行。

摄像头传感器和红外线过滤镜：鉴于我们的多点触摸技术是基于红外线的，因此我们需要知道的是我们选择的摄像头是否具有读取红外线的功能。一般情况下，生产商都会提及所用的摄像头传感器类型，通过这个类型我们可以找到相对应的参数表，这个参数表会告诉我们这个摄像头传感器在不同波长光下的敏感度特性，例如：SONY ICX098BQ CCD 传感器的参数表（图 1.2.2）。

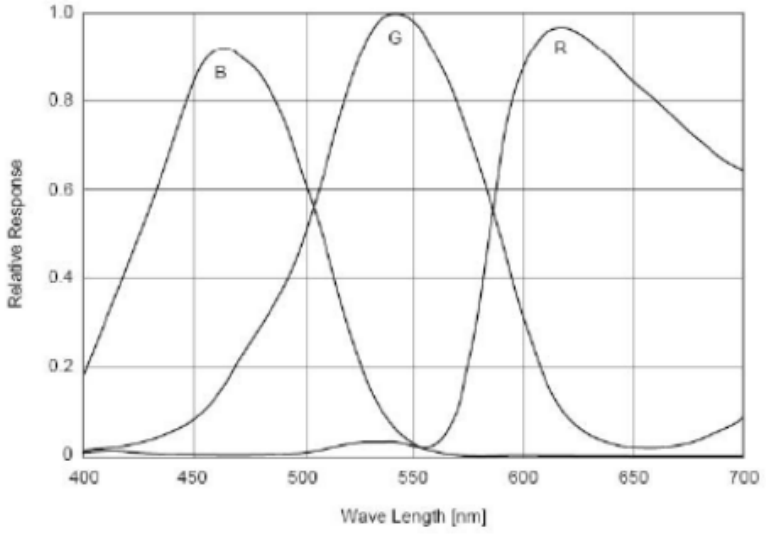


图 1.2.2　SONY ICX098BQ CCD 传感器

在我们利用摄像头作为我们的多点触摸设备部件前，仍需要为摄像头添加过

滤干扰光的滤镜。尽管我们使用的是已经能够感应红外线的摄像头，但它仍然会对其它光敏感，为了解决这个问题，我们需要在镜头前添加一个裁剪的滤片（软盘、三色胶片）或者镜头滤镜。裁剪的滤片能够为我们消去一些可见光，但没有特定的范围，而镜头滤镜具有波长唯一性，具有只允许一个特定波长通过的特点。

推荐的摄像头

如果是购买 USB 口的摄像头，我们建议您购买以下的一种：

PlayStation 3（PS3eye）摄像头，能够容易地除去红外镜片而且具有比较高的帧率（640*480 30 FPS）。

Philips SPC 900NC，不能够除去红外镜片，需要更换镜头但具有比较高帧率（640*480 30 FPS）。

如果是购买 IEEE 1394（火口）的摄像头，我们建议您购买一下的一种：

Unibrain Fire-i（640*480 30 FPS），用的传感器和 Philips SPC 900NC 的一样。

Point Grey Firefly（640*480 60 FPS）。

IEEE 1394 摄像头相对于 USB 摄像头具有以下的优点：

高帧率

大分辨率

图像压缩率低

1.2.3 显示设备

1.2.3.1 投影仪

投影仪是多点触摸中显示方式的方法之一，任何视频投影设备（LCDs，OLEDs 等）都可以，但是投影仪具有屏幕尺寸大和多功能的特点。

目前有两种投影仪类型：DLP（数码光处理技术）和 LCD（液晶投影技术）

DLP(数码光处理技术)是一种以成千上万微小的镜片反射光线实现投影的，利用色轮产生颜色。这种技术的投影仪字体对比度很高，而且投影仪的体积比较小。

LCD（液晶投影技术）是一种利用液晶的光电效应，即液晶分子的排列在电场的作用下发生变化，影响其液晶单元的透光率或反射率，从而影响它的光学性质，产生具有不同灰度层次及颜色的图像。这种技术的投影仪颜色非常的强烈，但在几年之后其颜色会逐渐褪掉。

对于投影仪需要考虑的是亮度，以 ANSI 流明作为衡量。数字越大意味着亮度越高，作为家庭影院，需要更高的流明，但对于多点触摸装置来说倒不一定。

当投影仪接近屏幕时，投影的尺寸就会变得小且亮度大，这样会引起局部区域过亮的问题，用户对着屏幕几分钟后会产生眩目的症状。

在投影仪中，最大的限制在于它的投影距离。要有一个适合的投影尺寸，就必须具有从投影仪镜头到投影屏幕之间的距离。例如：我们需要的投影尺寸是三十四寸，那么我们从投影机镜头到投影屏幕的距离是两英尺。因此在我们进行桌子大小设定的时候，需要对投影仪距离以及屏幕大小的设定进行规划。

在你制作一个箱子式的多点触摸装置时，镜子可以帮助你得到比较好的反射从而达到需要的尺寸。如果在设置一个比较空间比较紧凑的装置时，则需要一个具有短焦镜头的投影仪。

投影仪的是以宽高比作为衡量的，例如：如果你需要电视显示比例的，那么需要的对比率是 4：3。如果你需要一个宽屏的，那么需要的对比率是 16：9。在大多数情况下，都会以 4：3 的比例出现，但这在于所设定的交互界面。

1.2.3.2 液晶显示器

尽管投影仪在多点触摸装置中具有许多用处，但液晶显示器也能够为多点触摸装置提供显示的功能。液晶显示屏都是透明性质的，但液晶显示器的模型不是透明的。液晶面板本身是可以通过红外线的，将液晶显示器的所有部件拆除，保留 LCD 面板、电路控制板、电源，就可以用作多点触摸设备的显示部件，其显示效果与没有拆过的液晶显示器或者投影仪相当。

为了已拆解的液晶显示器能够正常显示，需要将液晶面板和电路控制板连接起来。面板和电路控制板是通过柔性扁平电缆（FFC）连接的，因此需要长度适合的柔性扁平电缆来确保在多点触摸装置中的放置。不同的液晶显示器具有不同长度的 FFC，在拆解液晶显示器时需要注意关于 FFC 的参数，当然我们也可以通过在电子商店里购买 FFC 来延长我们的长度。

1.3 受抑全内反射多点触摸技术（FTIR）

FTIR 的名字来源于 NUI 论坛，这种多点触摸技术的发明者是纽约大学的 Jeff Han 教授（Han 2005）。Han 教授的方法是源自一个光学的基本现象，叫全内反射（又称全反射），它讲述的是在入射角比特许的角（Getty,Keller and Skove 1989,p.799）大的情况下，光线经过两个不同折射率的介质，这个特许的角（称为临界角）基于物质的折射率而得到的，可以通过 Shell 法则以数学方式计算出来。

当上述情况发生时，在物质上就不会产生折射，而是所有的光线会反射在内部。Han 教授通过这个远离把红外线反射在一块遵守全内反射规则的压克力内部，当用户在压克力表面触摸时，光线就会被用户的接触部位反/折射（通过皮肤），在触摸的地方就会将原本反射在内部的红外线折射回我们在压克力板面架设的红外摄像头（图 1.3），通过对应的软件就可以侦测到我们相对应的信息点。

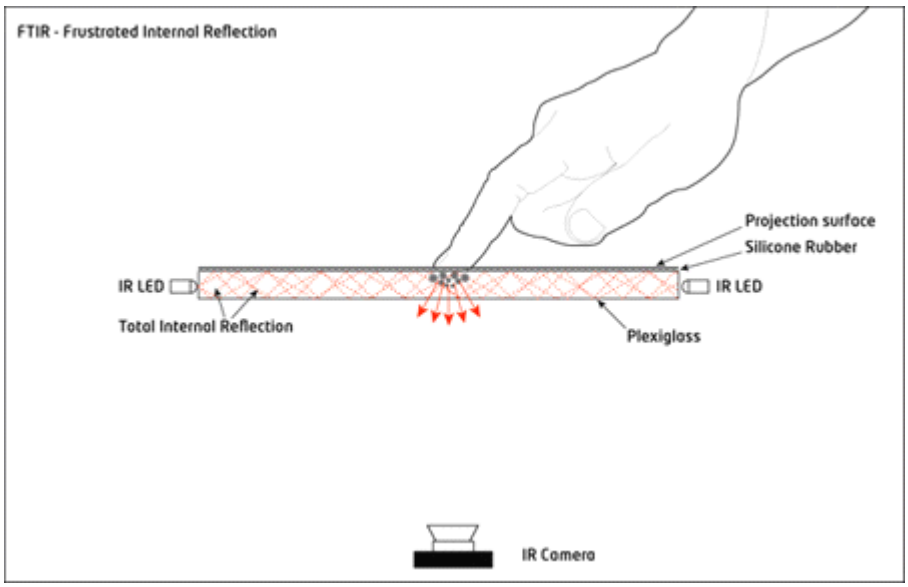


图 1.3 受抑全内反射多点触摸技术原理图

1.3.1 受抑全内反射（FTIR）层

这个原理在我们制作多点触摸显示设备的时候非常的有用，当光线的反

射被我们破坏在对应的区域，然后折射出来时，架设在下面的红外摄像头便能够清晰地读取到相对的信息点。

压克力

根据 Han 教授的论文，我们需要利用压克力作为我们的现实面，而这压克力的厚度是需要特定的，一般为 8mm，为了避免压克力变形，在大面积的情况下应该用 10mm 的。

在我们利用压克力作为我们的多点触摸显示面时，我们需要做一些前期准备，因为通常情况下我们购买的压克力的四边是粗糙的，我们需要进行抛光，这样能够让光线更好的从四边打进压克力内部。如果是自己抛光，我们将教大家用砂纸来进行抛光工作：先利用号数小的砂纸进行粗抛，然后逐级的上升，达到晶莹剔透的效果。

隔板

为了避免让光在射进压克力内部的时候从边缘处漏出来，因此利用隔板盖在压克力四周。隔板的材料没有限制，可以是木条或者铝条。

漫反射层

漫反射层的作用在于让摄像头不被其它物体或者光线干扰，只读取到非常光亮的点（手指触摸时产生的反射点）。

兼容层

一个基本的 FTIR 装置的运行效果取决于用户在触摸时手指的光亮度，潮湿的手指能够更好的产生对比度，触摸起来会更流畅，而干燥的手指或者物体则不能够产生破坏全内反射的效果。为了解决这个问题，我们需要添加兼容层在显示板的上面，这个兼容层扮演着一个代理的角色，能够提高我们破坏全内反射的效

果，兼容层可以用硅胶材料来制作（例如：ELASTONSIL M 4641）。为了提高以及保护触摸屏幕的敏感度，可以利用投影幕，（例如：Rosco Gray #02105），在这种情况下就不需要再放置漫反射层了。

1.3.2 兼容层

兼容层（兼容面）是用在 FTIR 装置上的一个层，能够提高 FTIR 装置的敏感度。它连接着压克力，当发生触摸事件时就会产生明显的 FTIR 效果，当没有触摸时则被释放。这个兼容层只适用于 FTIR 装置，DI、LLP、DSI、LED-PL 则不需要。兼容层可同时用作投影幕。

兼容层（兼容面）是简易放置在投影幕和压克力之间的层，它能够提高手指反射的对比度以及完整度，也能够避免手指粘附在表面的情况。在 FTIR 多点触摸技术中，红外光是从压克力四周射进内部的，光线在压克力内跑动（全内反射，就像光纤线一样），当你触摸屏幕的时候（破坏了全内反射）就会让光从压克力内跑出来，这就会产生对应手指的信息点（光亮的点）。

接下来会有更多关于如何做一个比较完美的兼容层：

找一些比较不错的材料，例如：Sorta Clear 40、类催化剂硅胶、Lexel、RTV 品牌的各种硅胶，不过也有人利用纺织布料来做，例如：硅胶浸渍和 SulkySolvy。

但最原始、最受欢迎而且容易成功达到效果的方法是直接喷洒一层光滑的硅胶层在压克力表面，待其干后在其上面再放置投影幕。这样需要的材料是接近透明的，喷涂上去后会成为压克力的一部分，需要让红外线能够穿透过去，因此需要用到先前提到的三种硅胶材料，它的折射率可以接近压克力。目前已经有人制作了专门给 DIY 爱好者的这一工序的配件，推出之后很受欢迎。但需要注意的是，在放置兼容层（硅胶层）的时候要将其放置在投影幕的下方，然后放在压克

力上，这样子会得到更多好的效果。

关于利用纺织物作为兼容层，很少具有比较成功的效果，但如果能够找到更好的材料，这将能够更容易地使用。目前很多人在寻找一种更好的兼容层，能够作为投影幕的同时也能够起到兼容层的作用，但在 FTIR 技术上，兼容层没有一个基准线，以下是关于兼容层的一些优点：

能够保护压克力，避免被用户在操作的时候刮花

阻挡了很多光线的污染

能够提供流畅度 (在触摸没有加装其它东西的压克力时，很容易有不流畅的感觉)

在触摸面和投影面上没有错落

提高压感

提高触点的对比度

为利用 LCD 作为显示屏的 FTIR 技术设置一个兼容层是十分困难的，因为这需要所用的材料能够十分的清晰且没有变形，以及不会影响到液晶屏的显示。这和投影仪作为显示屏不同，投影仪可以将画面直接投射在这个兼容层上。到目前为止，没有一个人能够为基于 LCD 为显示屏的 FTIR 技术设置一个百分百清晰的兼容层。尽管如此，也有很多爱好者成功的搭建基于 LCD 为显示屏的 FTIR 装置，但他们没有用到兼容层，这可以说明兼容层的需要是在于所需要的装置是怎样的。

1.4 散射光照明多点触摸技术（DI）

散射光照明多点触摸技术会有两种表达形式：正面散射光照明多点触摸技术和背面散射光照明多点触摸技术。两种表达形式都基于同一个原理——画面与触摸在屏幕上面的手指形成对比。

1.4.1 正面散射光照明多点触摸技术

可见光（通常指来自周围环境的光）照射在触摸屏幕的正面上，将漫反射幕放在触摸屏幕的上方或者底部，当物体触摸屏幕时便会产生阴影，而摄像头就会根据这些产生的阴影来读取信息点。

1.4.2 背面散射光照明多点触摸技术

在背面散射光照明装置上，需要根据装置的大小和形状独立地来分配红外照射器放置的位置。当某个区域容易产生触摸效果，那么其它地方就显得效果不明显了，这时用户需要用力触摸，使其产生触摸事件。为了解决这个问题，我们第一步需要硬件进行设定，例如：更换材料、重新摆放照射器的位置等。如果硬件装置无法解决或者让效果更好，那么我们需要用到软件来进行优化。

目前为止，Touchlib 提供了任何在相同强度下的整体图像的滤镜。可以通过滤镜来修改不同图像区域的强度来弥补不断变化的光线条件。利用梯度的方法来设定应用在每个像素强度的灰阶图上，从而定义某个特定像素的滤镜。这种灰阶图可以作为额外的校对步骤。

红外光从底部照射在触摸屏幕上，将漫反射幕放在触摸屏幕的上方或者底部，当物体触摸屏幕的时候会反射比漫反射幕更多的红外光，然后被摄像头读取（如图 1.4.2）。用这个漫反射幕也可以用来检测悬停和在界面上的物体。背面散射光照面多点触摸技术（如图 1.4.2 原理图）需要用到一个可以发射红外光的照

射器，这种照射器可以在市场上购买，或者通过自己来焊接独立的 LED。和其它多点触摸技术不同的是，背面散射光照明多点触摸技术需要某种特定的漫反射来把光漫射，这种材料也常能够作为投影幕。

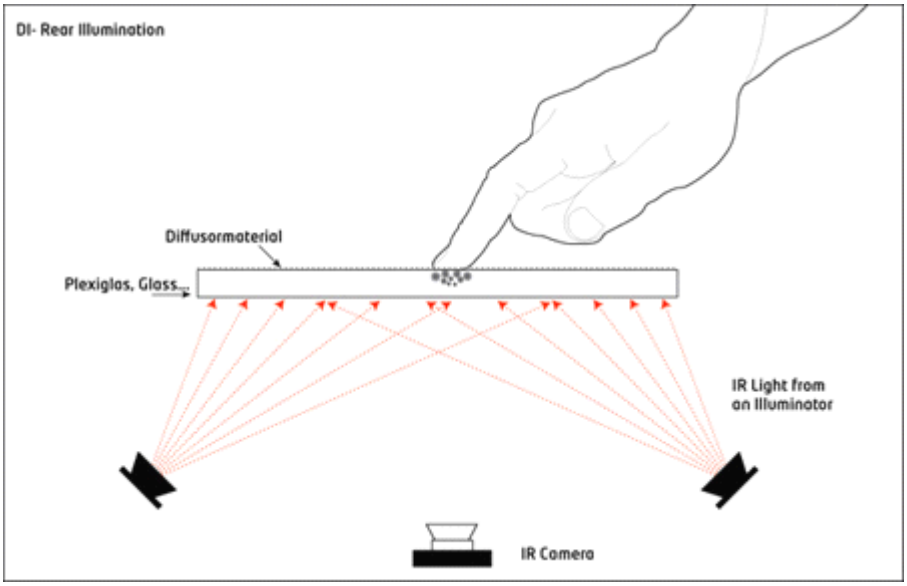


图 1.4.2 背面散射光照明多点触摸技术原理图

1.5 激光平面多点触摸技术（LLP）

红外激光头发射出来的红外面铺满整个屏幕，这个激光红外面的厚度大概在1mm 左右，当手指触摸屏幕的时候，手指的尖部会作为一个红外点显示出来（如图 1.5）。

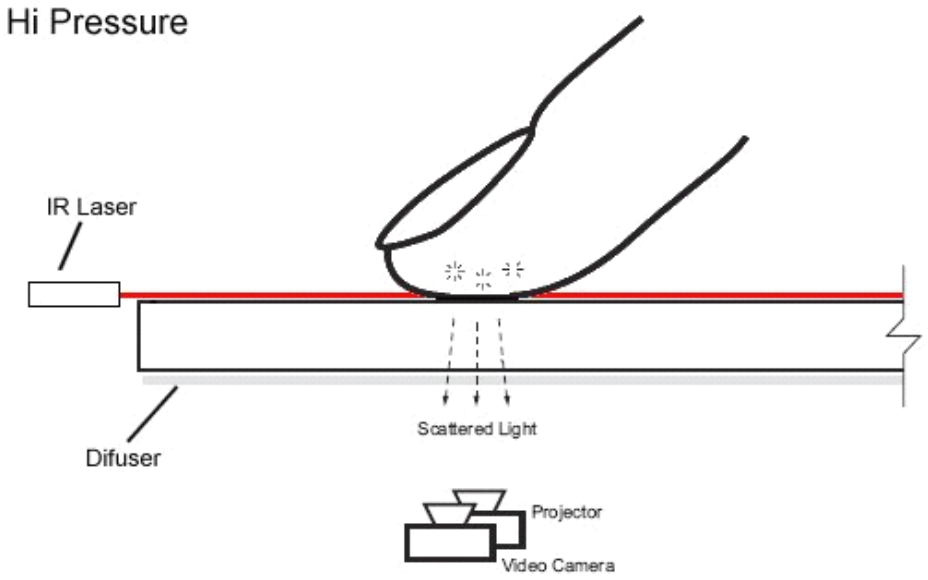


图 1.5 激光平面多点触摸技术原理图

利用激光平面多点触摸技术来实现多点触摸装置是种简易以及价格便宜的方法，多数的装置需要两到四个激光头，从屏幕的四个角落照射在平面上。激光光线的亮度取决于激光的功率（mw、w），功率越大亮度就越高。

通常用到的激光头波长会在 780nm 和 940nm，大部分爱好者都是从美国的 Aixiz.com 网站购买的，这种激光头需要一个可以让光形成一条线的镜头，120° 光线的镜头是目前为止最受爱好者欢迎的，因为可以减少激光的数量。

在运用这种技术时，一定要注意好激光头的安全性，学习好关于激光头的常识以及在设置的时候要注意激光头光线的方向。

1.5.1 激光头安全性

红外激光头是 LLP 效果的主要部件，但这些激光头会给我们带来一定的危险性。大多数 LLP 装置中，5mw-25mw 已经足够了。尽管处于这个级别，但在装置的过程中对眼睛仍然有危险性。可见光激光头，我们直视的时候同样具有危险性，但正由于其具有可见光，我们可以清楚的知道如何去避免这些伤害。但红外激光头不同，在具有伤害性的同时又是不可见的，我们无法琢磨到激光光线处于什么地方而会无意中伤到眼睛。下面是关于激光头级别分类的一段文字（来自维基百科）：

> *A Class 3B laser is hazardous if the eye is exposed directly, but diffuse reflections such as from paper or other matte surfaces are not harmful. Continuous lasers in the wavelength range from 315 nm to far infrared are limited to 0.5 W. For pulsed lasers between 400 and 700 nm, the limit is 30 mJ. Other limits apply to other wavelengths and to ultrashort pulsed lasers. Protective eyewear is typically required where direct viewing of a class 3B laser beam may occur. Class-3B lasers must be equipped with a key switch and a safety interlock.*

在附录 B 中解析到，线性镜头是用来让原本只有一点光线的激光变成一个面。线性镜头减低了激光的强度，但是其危险性仍然存在。因此在搭建 LLP 装置时，十分有必要佩戴一个相对应波长的保护镜。在 LLP 装置里，不能放置会引起反射的物体，例如：一个酒杯的圆柱底部会引起激光无序地散射出来。在装置中，为了不让光会反射出来或是避免用户能够直接看到激光头，应添加能够阻挡激光线的隔板。激光保护镜中会根据不同的参数有不同的保护范围：光密度是针对不同波长和不同镜片的衡量值。是以一种对数方式出现，例如：光密度值为五的能够削减光线的一百倍，这比光密度为三的要大。在设置激光角度的时候还需要用到一个低功率的可见激光器，因为它的安全性以及可见性可以减低我们在

设置红外激光时的危险性，设置中我们利用“检测卡”（可以来检测可见光激光线的卡或者盒子）来校对激光线，这对我们在设置的时候非常有用。

再次提醒要注意相关的安全预防措施：在设置红外激光的时候要一直戴着红外激光保护镜，不要用激光直接照射自己的眼镜。如果能够遵循这些措施，那么 LLP 会是一个安全的装置，否则这对自己的视网膜造成严重性的伤害。

1.6 散射光平面照明多点触摸技术（DSI）

散射光平面照明多点触摸技术是利用一种特殊的压克力来使红外线照亮整个屏幕的。可以参照 FTIR 装置的步骤（不需要用到兼容层），把普通的压克力转成特殊压克力(如图 1.6),这种压克力运用了许许多多的导光粒子作为主材料，就像在压克力里面装满了许许多多的镜子一样，当光照射进入内部的时候，就会被发射从而照明整个屏幕，市场上称这种压克力为导光板。这种效果有点类似散射光照明多点触摸技术（DI），不同的是没有特别明亮的区域以及它的设置和 FTIR 相同。

导光板的型号有几种，这是根据不同的厚度和不同的导光粒子数量来定义的。目前，市场上的导光板厚度是 6mm-10mm，以“L”、“XL”、“XXL”三种型号来区分内部导光粒子的数量。一般情况下，6mm 厚度的导光板对于桌子设置来说有点柔软，10mm 是最好效果的。

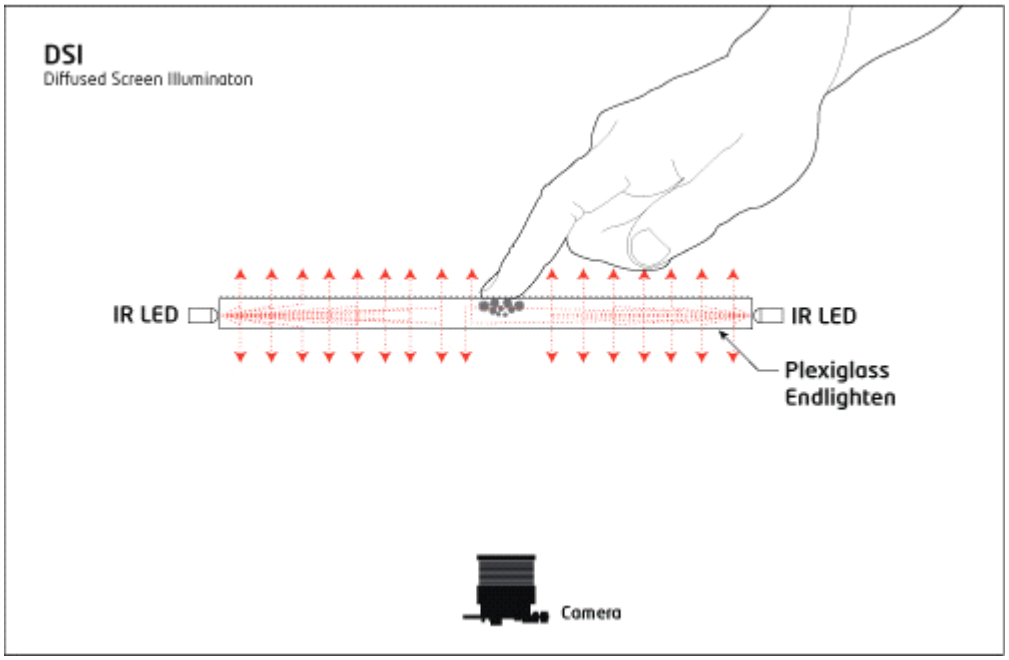


图 1.6 散射光平面照射多点触摸技术原理图

1.7 发光二极管平面多点触摸技术（LED-LP）

发光二极管平面多点触摸技术的设置和 FTIR 一样，不同的是压克力的厚度以及红外光照射的方式（红外光从四周照射在压克力屏幕表面上），下图 1.7 是发光二极管平面多点触摸技术的示意图。

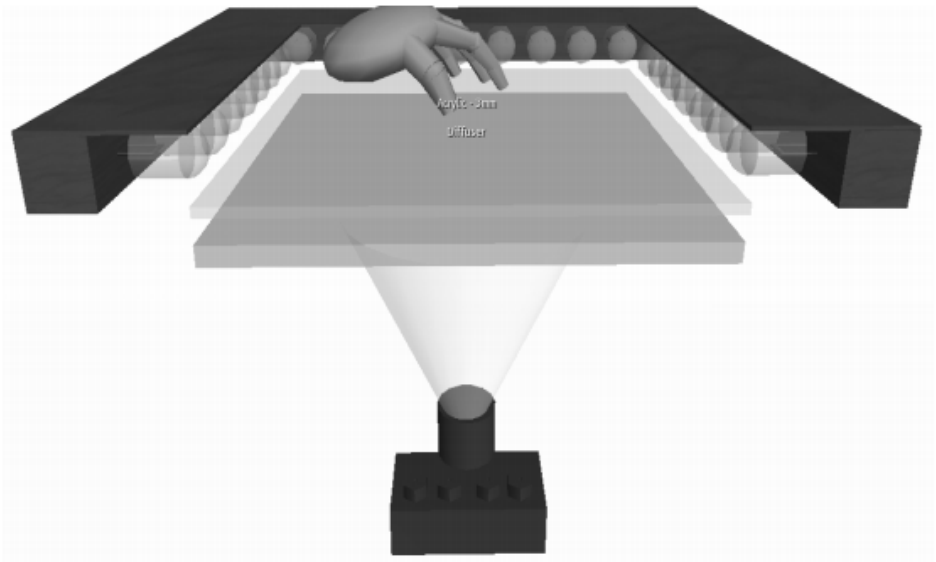


图 1.7 发光二极管平面多点触摸技术示意图

红外发光二极管放置在触摸屏幕的四周，让光线更好地分布在表面上。这和激光平面多点触摸技术类似，二极管平面多点触摸技术同样在触摸屏幕上创造了一个红外线平面，光线会使放在屏幕上方的物体发亮而不是触摸，然后通过软件（Touchlib、Community Core Vision）调节滤镜来设置仅当物体被提起或者接近屏幕的时候被照亮。但这个对于刚开始用二极管平面多点触摸技术来做设备的爱好者来说会是一个挑战，需要有更多的耐性去调节。装置中建议利用一块挡片（例如图 1.7 所示）放置在二极管的上方，这可以让更多的光在平面上。

一般情况下，我们推荐当利用液晶屏作为显示屏幕的时候运用二极管平面多点触摸技术（LED-LP），和散射光平面多点触摸技术（DI）以及激光平面多点触摸技术（LLP）一样，触摸屏幕不要像 FTIR 那样的厚度，但需要能够承受用

户使用时产生的压力。

1.7.1 运用的层

第一，触摸屏幕需要一个坚硬的可以抵抗用户交所产生的压力之外，还需要一个清晰可见的屏幕，一般会用压克力或玻璃作为触摸屏幕。如果是用投影仪来作为显示的，则会用到投影幕，如果是用液晶（LCD）作为显示屏的则需要在液晶屏（LCD）的下方放置一块漫反射层来避免液晶背灯光的干扰。

对于发光二极管平面多点触摸技术（LED-LP）的红外光源最少需要从触摸屏上方的两边发射出来，通常情况下，越多的边发射光源，则能够得到越好的红外线环境。

图 1.7.1 发光二极管平面多点触摸技术（LED-LP）装置

1.8 各种技术的对比

对于爱好者，通常会问：“什么样的多点触摸技术是最好的呢？”很抱歉地告诉大家，这个问题没有一个直接的答案，因为每个技术都有自己的优点和缺点。下面的对比也许能够告诉大家什么是最好的或者什么是对于大家的需求来说是最适合的，而这答案只是针对你的。

受抑全内反射多点触摸技术（FTIR）

优点	缺点
不需要一个封闭的箱子 触点具有很高的对比度 允许各种点的压感 具有兼容层，可以读取到笔尖那么小的点	需要焊接发光二极管（LED） 需要兼容层（硅胶层），不能够用玻璃作为显示幕 无法识别物体的标示

背面散射光照明多点触摸技术（Rear DI）

优点	缺点
不需要兼容层，仅需要投影幕或漫反射幕 能够用玻璃或者其它透明的材料 不需要焊接发光二极管（LED），可直接购买红外发射器 能够识别物体或者其它标示	难以取得一个好的照明方式 触点的对比度低（软件难以识别） 具有较多无法识别的触点 需要一个封闭的箱子

正面散射光照明多点触摸技术（Front DI）

优点	缺点
不需要兼容层，仅需要投影幕或漫反射幕 能够用玻璃或者其它透明的材料 不需要焊接发光二极管（LED），可直接购买红外发射器 能够识别物体或者其它标示 不需要封闭的箱子 容易搭建	难以取得一个好的照明方式 不能够识别物体或者标示 具有较多无法识别的触点 不太可靠（环境的影响很大）

激光平面多点触摸技术（LLP）

优点	缺点
不需要兼容层，仅需要投影幕或漫反射幕 能够用玻璃或者其它透明的材料 不需要焊接发光二极管（LED） 相对便宜 不需要封闭的箱子 容易搭建	没有真正的压感 不能够识别物体或者标示 如果仅使用一到两个激光头，会容易引起触点阻挡的情况

散射光平面多点触摸技术（DSI）

优点	缺点
不需要兼容层，仅需要投影幕或漫反射幕 能够轻易地从其它技术转换过来 能够识别物体或者标示 具有压感 没有局部区域过亮	导光板相对普通的压克力，价格要高 与 FTIR 和 LLP 相比，触点对比度低（软件难以读取） 具有较多无法识别的触点 尺寸因为其柔软度受到限制

发光二极管平面多点触摸技术（LED-LP）

优点	缺点
不需要兼容层，仅需要投影幕或漫反射幕 能够用玻璃或者其它透明的材料 相对便宜 不需要封闭的箱子 不需要 LED 框	需要焊接发光二极管（LED） 不能够识别物体或者标示 不能够用 LED 带

第二章

软件和应用程序

本章预览

2.1 软件编程导论

多点触摸输入编程和其他任何形式的编程一样，不过在多点触摸编程里，有一套自己的某些协议，方法和标准。通过 NUI Group 与其他组织和社团的工作，多点触摸编程已经有了针对多种编程语言的开发框架，这些语言包括 ActionScript 3，Python，C，C++，C#以及 Java。

多点触摸编程分为两步：首先，从摄像头或者其他输入设备读取和转化输入的触点信息，传递这些原始的触点信息通过之前制定的协议组合成手势，然后高级编程语言就可以使用手势来让一个应用程序配合。TUIO(Tangible User Interface Protocol,可触摸的用户界面协议)已经成为追踪触点信息的行业标准协议。

以下章节将讨论多点触摸软件的两个方面：触点追踪和应用程序运行框架。

2.2 触点追踪

对象追踪一直是计算机视觉基础研究领域的一个方面。它的工作是跟踪包括能够准确的反复识别包含特定对象的一系列视频帧（估算）。一般来说，这是一个非常困难的问题，因为首先要在所有的帧中发现对象(而且往往是在杂乱，封闭，或者是不断变换的照明条件下)，以及让数据能够和帧之间以某种方式联系起来以便识别我们所需要的对象。

现在很多的问题已经被解决，在追踪这个问题上最普遍的模式就是“生成模式”(Generative Model)，这是一些诸如 Kalman 粒子过滤器等流行解决方案的基础。

在大多数的系统中，一个完善的背景相减算法需要对每帧进行预处理，这确保静态或者背景图像能够被忽略掉。对于一些光照不稳定的视频流，像“高斯混合模型”(Gaussian Mixture Model)这样的自适应模型已经能够比较智能的识别出不均匀的动态背景。

把背景过滤掉之后，剩下就是我们需要的前景对象了。我们往往确定这些对象的质心，而且这些点会被一帧一帧被追踪。追踪算法会根据这些萃取的质心估算在下一帧触点的位置。

稍稍说一下这个追踪，例如，一个基于 Kalman 过滤器的简单模式可能是一个线性恒定速度模型。一个 Kalman 过滤器有两个动力学方程，一个是 STATE 方程(“状态方程”)，另一个是 OBSERVATION 方程(“观察方程”)，在这个例子中，

$$x_k = x_{k-1} + Vu_k + w_k$$

这套方程分别是 $y_k = x_k + n_k$

该状态方程描述了在有噪声的情况下状态变量‘xk’的变化情况。如果我们说‘xk-1’表示物体在‘k-1’时刻的真实位置，那么这个模型就可以基于一定的速度、‘Vuk’和噪音因素，根据线性组合的方式，推测出‘k’时刻物体的位置。到目前为止，根据这个模型，我们还不能直接的通过观察预测物体的位置 Xk，我们需要观察方程。观察方程用来描述实际观察变量‘yk’，在这里，‘yk’被定义为真正位置‘xk’的一个噪音观察值。噪音的值也假定是均值为零的高斯白噪声。运用这两个方程，卡尔曼滤波器就能根据物体的上一个位置，通过递归的方式预测出物体的位置。

每个状态的预测，数据追踪模块都会去按照事先的预测和观测数据去追踪对象。如果没有找到符合预想的对象，那么这个对象就会被视为一个新的对象而被追踪。

2.2.1 多点触摸的追踪

追踪是多点触摸非常重要的技术。它允许多个手指同时进行控制。我们也可以识别出手势，因为每个手指的轨迹都可以被记录并输出。没有追踪技术，这是不可能的。

值得庆幸的是，今天的多点触摸硬件大大地简化了追踪对象的流程，因此，即使是最简单的 Kalman 滤波器实际上也成为不必要的了。现在的追踪系统性能瓶颈趋向于怎么产生并维持一个背景模型。大量的计算会使 CPU 严重超负，除非出现更加智能的方案。然而现在的基于红外线(IR)的硬件方案，比如 FTIR 或 DI，自适应背景模型(adaptive background model)会被扼杀掉。由于非红外光几乎都会被过滤掉，所以大部分的背景就被硬件给删除了。为了这些红外图像，捕捉一个静态的背景图往往会删除几乎所有的环境光。这个背景图像会减去其后所有的帧，剩下的这些帧将作为阈值应用给系统，然后我们就只剩下了那些突起的点，

这些点也就是我们想要捕捉的手指或者表面对象，称之为“触点”。[译者注：这个可能比较难理解，我给大家举个例子吧，比如我们在 PS 中的“阈值”命令，可以将灰度或彩色图像转换为高对比度的黑白图像。“阈值”命令可以确定图像的最亮和最暗区域。想想我们的在 CCV 或者 tbeta 中看到的捕捉后的图像，不就是黑白对比非常明显的图像么？背景是纯黑，手指尖是纯白。]

此外，只要给出范围，那么追踪问题就比较简单了。我们知道，从一帧到下一帧，以标准的 30Hz 计算的话为 33ms，一个人的手指在这段时间里移动的距离非常有限。根据这种预测，我们不用研究物体的动力学，只需要找两帧之间最近的匹配物就是了。近邻比较的方式是比较相应的数据，一般是比较欧几里德几何距离。通过这种方式，一个数据点和一系列靠近它的‘k’点进行比较，得到该点的新位置。通过这种方式，我们可以比较可靠的追踪某个具体的触点。试误法是经常采用的方法，当情况不确定的时候，我们总是试探着处理这些情况，比如，当一个物体挡住另外一个物体的时候。

在 Touchlib 和 CCV 框架中都有追踪器的实例。使用 OpenCV，一个开源的计算机视觉库，可以直接处理图像和视频流，能够非常准确地实时追踪到触点。当一个触点被发现，消失或者移动的时候，这些触点的相关信息(位置，ID，区域等)都会以事件的形式发送出去。开发者就可以利用这些信息去监听这些触点开发出应用程序。

2.3 手势识别

NUI 的挑战 : “Simple is hard.Easy is harder.Invisible is hardest.”——Jean-Louis Gassée

2.3.1 手势的图形用户界面

未来的人机交互将是自然用户界面(Natural User Interface),当然这个界面还很模糊。随着便宜又可靠的多点触摸硬件的不断发展, 我们相信, 在不久的未来 , 多点触摸设备不仅仅是在实验室里了,而是遍布在学习室,绘图室甚至是厨房里 , 无限可能。

鼠标和图形用户界面一直是电脑可以在社会上大规模普及的重要原因。然而, 传统的人机交互是间接的和需要识别的一种方法。建立在多点触摸上的人机自然交互是直观的，流畅的和令人回味的。基于手势的 GUI 界面将进一步使计算机成为我们生活中的不可或缺的一部分。

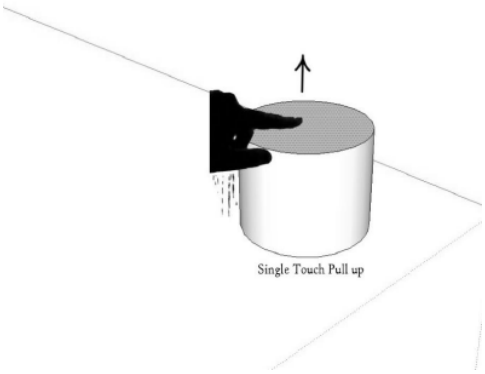
从广义上说，“手势的概念涵盖很广泛，只要是为了让交流的目的更明确，更引入瞩目而采用的一切身体动作，都可以称作手势。”通过手势交流是人类发展历程中最古老的形式，当然，这超出了我们的讨论范围。GUI 系统是利用人们以往的阅历和认知来熟悉应用程序，而 NUI 的界面充分利用了人们的设想和合乎情理的结论从而提出了一个直观的、基于手势的内容界面。这样一个基于手势的界面是社会化，任务协作，以及艺术性触摸的最佳候选方案。这个界面是符合自然规律的、更直观的用户界面。

然而, 现在多点触摸硬件上所使用的手势是很少的, 仍然存在很大的发展空间, 当然也少不了继续探寻一些手势的可应用性。多点触摸界面需要一个全新的方案, 而不是用 GUI 或者 WIMP 的方法来实现它。手势的类型决定了不管是多

触摸用户还是单触摸多用户都能进行互动。我们将讨论需要的新手势，发展手势识别模块和支持框架能够充分利用多点触摸设备的潜力，支持可定制开发和易于使用的复杂的多点触摸应用。

2.3.2 手势 Widgets

NUI 的多点触摸方案提供了一套强大的手势界面平台，而且是基于对象的(而不是 WIMP)，从而消除了现实世界与应用程序之间的抽象化。界面的目的在于让人们能够直接操纵、更好的沉浸式体验和容纳界面上的低精度操作。流行的手势诸如用两个手指将图片缩放、旋转，这被称之为“操控手势”，也是自然手势的一个很好的例子。



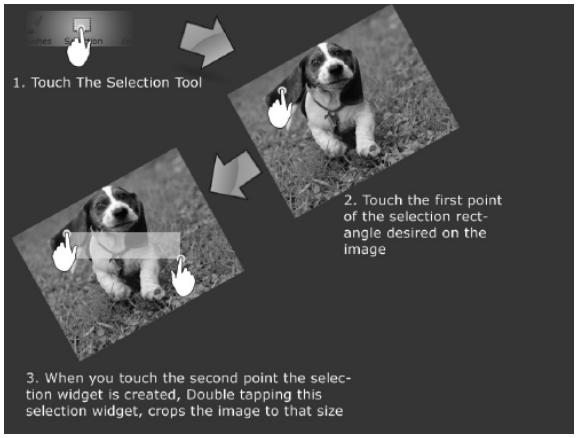


图 1：多点触摸程序中的手势范例

新类型的手势 widgets[图 2]对对象的直接操控要有一个全面的概念(在 NUI 的概念中，一切都是对象)，有一个流畅且具有唤起性(evocative)的环境，不仅仅

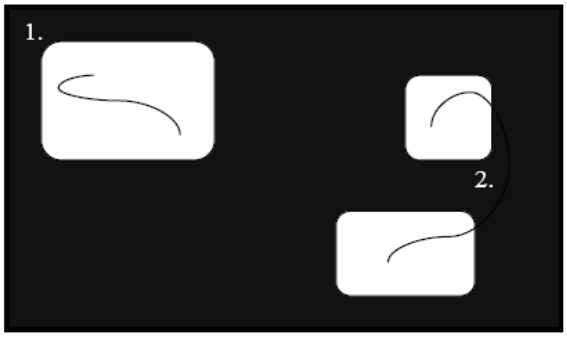
是模拟鼠标。

图 2

Gesture Widgets 应该具有创造性思维，能够存储用户的反馈并进行处理。那么这些 Gesture Widgets 就可以被扩展来设计开发更复杂的应用，它们还应该支持自定义手势。

2.3.3 手势识别

手势识别研究的主要目的是建立一个能够识别人的特定的手势然后发送这些信息或者直接控制设备的系统。为了确保能够比较准确的识别手势和呈现一个直观的界面，在这个系统模型上会做一些限制。多点触摸设备应该提供一个以上的用户独立工作或者协同工作的能力，然后让多点触摸应用和多种多点触摸设置模式在一起工作。



Touch event 1 sequence is single gesture while touch-event 2 will be sent as three gestures.

图 3：手势示例

手势(Gesture)一词被定义为起点和终点在一个范围内，而且在两点之间有动作。在多点触摸输入的时候，系统应该能够识别在空间和时间内有意义的手势。手势识别程序可分为三个渐进的过程。

检测意图：手势仅需要在用户操控应用程序窗口的时候解析，X 延展输入协议以及即将到来的 X12 会通过 X 服务器来担当继续地转述触摸事件的责任，通过软件的执行正确地反馈给应用界面。

手势区分：在同一个应用程序操作触摸事件的时候会有同样的一套动作，因此，触摸事件需要基于对象意向情况下被重新定义，这些被重新定义的数据会被发送给手势识别模块。

手势分类：手势识别模块会将被定义的数据映射到相对应的命令中，这会有各种手势识别能够用到，可以是单独的或者像 Markov Models, Artifcial Neural Networks, Finite State Machine 等那样整合起来的。

其中最重要的技术是隐形马尔科夫模型(Hidden Markov Models,HMM)。它是根据历史数据，预测等时间间隔点上的各类对象分布状况。此方法的基本思想是根据过去对象变动的规律，推测未来对象变动的趋势。

一个仿真的神经网络(artificial neural network, ANN), 通常只叫做 NN(neural network), 它是一个建立在生物神经网络理论上的计算机模型[图 4], 它在多变的环境中非常灵活。新的方法可以同时使用 HMM 和较为常见的 ANN 来识别对象。

手势识别模块还应提供联机和脱机识别功能，以及 shot 和 progressive 的手势。依照现在的观点，可以表示为:

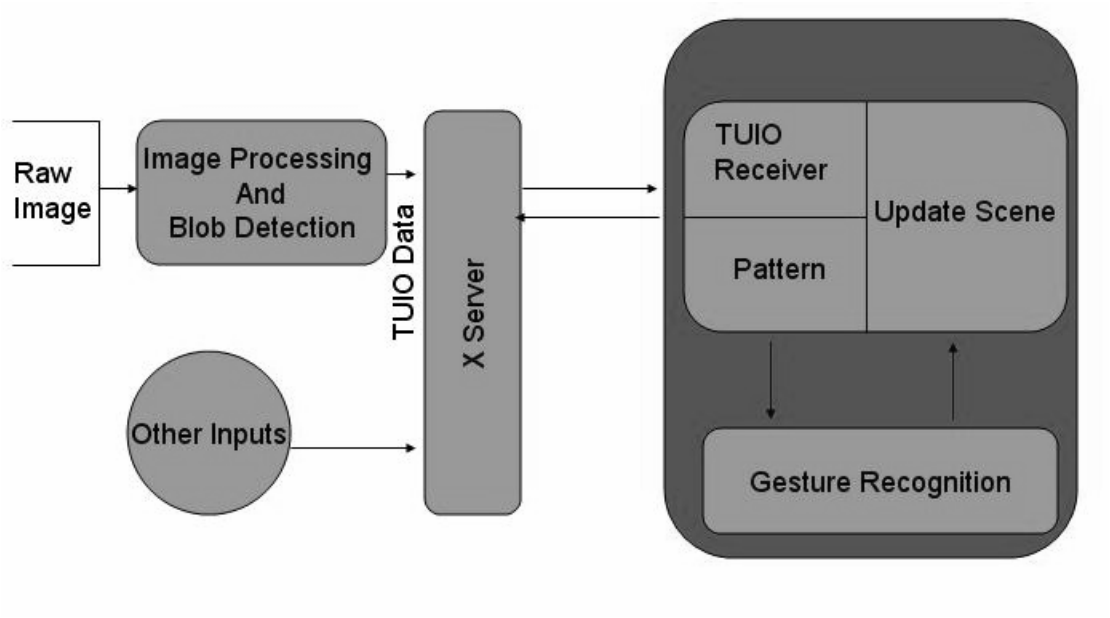


图 4: 在离线模式下，触点追踪和手势识别的框架

从客户端架构分离的手势识别模块本身就可以构成 MVC 架构, 这样有助于稍后讲到的语音识别模块和其它的手势识别模块以适当的形式映射到界面控制器，然后更新视图和模型，使之成为更强大的界面。

2.3.4 开发框架

现在有一些框架已经发布了, 它们正在制定帮助多点触摸应用发展的触摸事件界面管理方式。然而，现在设备输入管理的方式仍然是抽象的，触摸事件通过 TUIO 发送（TUIO 是不考虑底层硬件的)。而手势识别能真正地挖掘多点触摸界面的潜力，这也是客户端的任务。现在一些通常的手势，特别是那些操控图片

的手势已经出现了。这些框架不是侧重于手势界面，它们更侧重于发展多点触摸环境的 GUI 和 WIMP 的标准接口。

手势和手势识别模块随着 NUI 界面的推出已经有了长足的发展。一些重要的框架有：

Sparsh-UI

(http://code.google.com/p/sparsh-ui/)

Sparsh-UI 是一个发布在 LGPL 许可证下的多点触摸手势识别框架，貌似也是第一个关于手势识别的框架。它可以连接各种硬件设备，并且可以跨平台使用，同时支持不同的编程语言和 UI 框架。触摸信息从连接装置捕获之后会进行手势的处理和识别，每一个在客户端的可视化组件都可以被“联想”成一组特定的手势，然后试图去识别它。新的手势和识别算法可以添加到包中设为默认。

Gesture Definition Markup Language

(http://nuigroup.com/wiki/ Getsure_Recognition/)

GDML 是基于 XML 的一种用来说明在输入界面上如何建立清晰的手势事件的标记语言。通过 XML 来描述手势的状态，同时支持在指定的基础上创造新的手势，这也就是说，手势引擎支持自定义的手势。

这个项目设想成两部分：

描述一个标准的手势库(Gesturelib)来满足大部分应用的需要。

类库代码支持已经既定的手势，并且可以将手势事件发送给应用层。

Grafiti

Grafiti 是一个建立在 TUIO 客户端上的 C#框架，可以用来管理桌上界面多点触摸的交互行为，也支持一些既定的有形物体的触摸。这个项目的目的是支持

使用第三方模块来加强(专业)手势识别算法。当然，一些基本的手势已经包含在里面了。

NUIFrame

NUIFrame 是一个基于上述讨论模式的 C++框架(目前正在开发中)。它提供了一个单独的手势识别模块，除了处理一连串触摸事件之外，NUIFrame 也提供了客户端程序。这就可以确保不同的对象在使用相同手势的时候，可以根据背景做出不同的运算。它还将支持基于用户特定的规格来自定义手势。这套手势工具支持图形化的自动调试所生成的手势。

AME Patterns Library

AME Patterns Library 是一个新的 C++模式识别库，目前的重点是实时手势识别(real-time gesture recognition)。它使用基于概念的程序设计来表达通用的手势识别算法。这个库最近作为 AMELiA (the Arts, Media and Engineering Library Assortment，一个开源的库集)的一部分发布在了 GNU 通用公共许可证下。 它同时实现了两个多点触摸识别的隐形马尔科夫模型，同时减少了一些参数模型，简化需求条件，改进了运行时间性能，同时保持了良好的识别效果。

这里也有手势识别软件在准确性和实用性上存在的挑战：

手势识别中的图形噪声

不同的人在同一手势上存在差异

在给定的区域里找到普遍性的手势而不是去绘制难以记忆的手势

包容手势的可变性和复制性

慎重考虑与 GUI 系统相比有冲突的地方

Distinguishing intentional gestures from unintentional postural adjustments-

known as the gesture saliency problem.

悬空手臂式操作（gorilla arm）：人类并不习惯于在操控行为的时候将手举在前面，在通过一段时间的操控后，手臂回变得酸痛、局促、浮肿等，就像一只大猩猩一样在触摸屏上操作着。尽管短时间的操控没有什么问题，但从 1980 年早期开始将垂直式触摸屏作为主流输入技术，却阻碍了发展。

可以进一步改善 NUI 界面，比如在手势识别模块上加入压力感应，语音输入，面部识别等。

2.4 Python

Python 是一种可用于多种类型软件开发的动态面向对象编程语言，它提供了强大的与其它语言和工具相互协作支持，拥有广泛的标准库，而且你可以在几天之内上手。很多 Python 程序员都反映使用 Python 获得了更高的生产力，更强壮的代码以及更易维护的特性。

2.4.1 导读

这一节来阐述两种伟大的技术：多点触摸用户界面和 Python 编程语言。重点讲一下如何用 Python 开发多点触摸应用程序。

现在介绍一下 Python 里面的一个模块 PyMT 以及演示一下它的范例，然后尝试去写一下多点触摸输入的小程序。PyMT 是一个开源的、用于快速开发多点触摸应用程序的 Python 模块。

在许多方面，Python 和多点触摸在使电脑更容易操作这个方面用诸多的相同点。虽然作为一种编程语言，Python 需要更先进的技术，来让开发者能够快速而又轻松地编写代码。Python 常常说比其它语言更容易学习，许多 Python 程序员也反映使用这种语言使他们更专注地解决问题、实现目的，而不是拘泥于繁杂的语法和严格的语言属性。Python 强调其代码的可读性和可以从开源社区获取大量的可用模块，Python 试图用简单和有趣的编程方式构建多点触摸用户界面，来让电脑变得更加自然和直观地使用。

虽然多点触摸的实用性得到了快速的发展，但是就目前的互动方式和应用来看，多点触摸还有极大的潜力等待我们去挖掘。尤其是现在很多的多点触摸项目仍然处于在实验室内的阶段，所以能够最大限度的快速构建多点触摸交互和应用蓝图是非常重要的。Python 的动态语言特性和快速开发的能力以及社区里的大

量模块，都使得 Python 成为快速开发多点触摸交互和应用的理想语言。

2.4.2 Python 多点触摸的模块和项目(Modules and Projects)

以下简要概述 Python 里面涉及到多点触摸的模块和项目。

PyMT

PyMT 是一个用于开发多点触摸的 Python 模块，可以与 OpenGL 通信。它最初是一个爱荷华州大学的研究项目，最近在很多团体和个人的大力支持和共同开发下，PyMT 一直保持着较快地发展。它可以运行在 Windows，OS X 和 Linux 操作系统下，PyMT 的许可证是 GPL License。我们将在下面的章节中详细探讨 PyMT。

touchPy

Python 框架是与 TUIO 协议协同工作的。touchPy 监听 TUIO 的输入，并执行观察员模式(Observer pattern)，开发人员可以使用其子类。观察员模式使得 touchPy 平台和模块并不可知，例如它并不关心是哪一个框架来绘制屏幕。Alex Teiche 为它写了一个很重量级的教程。

PointIR

PointIR 是在 2008 年 PyCon 上演示的多点触摸系统，基于 Python。虽然当年看上去非常有希望，但是最近似乎从网络上消失了(搜索试试？)。授权信息也不清楚。

libAVG

根据官网的描述：“libAVG 是一个高层次的媒体开发平台，专注于互动装置。”虽然从严格的意义上讲，它不算是多点触摸模块。但是它却能够接收 TUIO 的信息和追踪触点，所以可以作为多点触摸系统来使用。libAVG 可以在 Mac OS

X 和 Linux 上运行，它也是开源的，许可证是 LGPL。

pyTUIO

一个用于接收和分析 TUIO 输入的 Python 模块。在 MIT 许可证下发布。

2.4.3 PyMT

PyMT 是一个用来开发多点触摸富媒体 OpenGL 应用的 Python 模块。主要目标是使开发变得更新奇，简便和快速自定义界面。本节将介绍讨论 PyMT 详细的结构，并使用它来作为一个例子讨论一些参与编写多点触摸界面的若干问题。

每一个有用的程序都做过两件事：获得输入和提供输出。如果没有输入，程序就只会输出相同的结果(例如"Hello World!")，那么这个程序是没有用的。如果没有输出，那么没人知道程序在做什么，或者根本就没有做什么。运行在多点触摸显示屏上的程序，最主要的输入方式就是触摸。图形的显示就是其主要输出。PyMT 尝试使输入和输出变得简单和灵活。在处理输入方面，PyMT 将 TUIO 协议包装成一个事件驱动的框架，输出方面则是基于 OpenGL，这样的话就可以使用图形硬件加速，在绘制图形上获得最大的自由性。

2.4.4 PyMT 的构造

图 5 展示的是 PyMT 的主要结构，如前面所讲的，它使用 TUIO/OpenGL 作为输入/输出。当前版本的 PyMT 依靠 pyglet，这是一个跨平台的 OpenGL 窗口和多媒体 Python 库，特别是 pyglet 的多媒体功能使得处理图像、音频和视频变得非常容易。大多数的媒体文件可以装载至单线(single line)的 Python 代码中(也包括绘制的 OpenGL 内容的重现)。

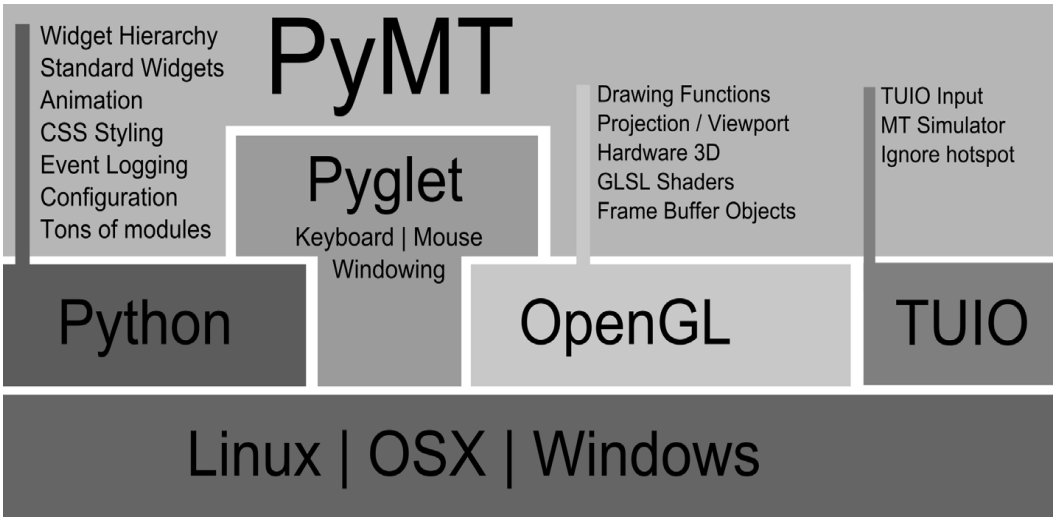


图 5: PyMT 结构。PyMT 建立在 Pyglet 基础之上，其中规定了 OpenGL 的窗口和多媒体加载的各种文件格式。它同样通过 TUIO 客户端监听输入信息。PyMT 将这些技术组合在一起，提供了一个面向对象的构件库和分级系统布局和事件传播。

2.4.5 OpenGL

利用 OpenGL 作为绘图后端利弊并存。虽然它可以在 2D 和 3D 渲染上拥有高性能和很好的灵活性，但是在毕竟处于比较低级的状态。而且它的学习曲线比较陡峭，也需要有基本的计算机图形学知识。

PyMT 试图抵消需要比较高级的知识才能驾驭的 OpenGL 绘图功能，提供了基本的绘图功能。PyMT 包括 drawCircle、drawRectangle、drawLine 和 drawTexturedRectangle 函数以及其它无需高级 OpenGL 知识的功能。使用 OpenGL 作为基本的渲染引擎，可以使高级用户充分控制他们的程序的可视化输出。PyMT 也提供了辅助函数和类去帮助实现高级的 OpenGL 编程。例如，PyMT 可以用一行程序实现建立一个帧缓冲对象(Frame Buffer Object)和从 glsl 着色。PyMT 还能利用引擎来处理投影矩阵和布局转变，这样就可以在不用调节他们参数的情况下直接运用。这个构思的原因是希望能够在大多数情况下能够最大限度

的方便运用，但是如果需要用到更高级的技术或者需要自定义相关技术则必须要用到 OpenGL 来编写。

对 OpenGL 的描述还有其它更加详细的资料，关于 OpenGL 的简单信息可以从[12][13]查阅到，标准的参考书是“The Red Book”，或者从 nehe.gamedev.net 网站上查阅经典的 OpenGL 指导。

2.4.6 Windows、Widgets、Events

大多数图像用户界面的开发包和框架都有一个叫作“工具（widget）”的概念，一个“工具（widget）”可以在同一个图形用户界面中构建模块，它是一个富有交互性及可视化的元素。在维基中是如下定义的：

> *In computer programming, a widget (or control) is an element of a graphical user interface (GUI) that displays an information arrangement changeable by the user, such as a window or a text box. [...] A family of common reusable widgets has evolved for holding general information based on the PARC research for the Xerox Alto User Interface. [...] Each type of widgets generally is defined as a class by object-oriented programming (OOP). Therefore, many widgets are derived from class inheritance.*
>
> *Wikipedia Article: GUI Widget[16]*

PyMT 和其它 GUI 工具板一样使用类似的概念，它以类似框架式的一部分提供一系列可以用于多点触摸的工具。但 PyMT 的重点在于让程序员很容易地支持自定义工具以及尝试开发新的互动技术，而不是提供一套标准的工具。这是来自如下建议和设想的主要的动机：

- 在用户自然界面（NUI）里面只有极少数的“工具”和交互设计应用证明了自己是标准的。
- NUI 本身就不同于传统的 GUI/WIMP（窗口、图标、菜单、指向设备）

- NUI 是非常具有内容化的，例如：可视化信息和交互应用是基于用户交互背景的。
- 传统的鼠标键盘系统已经无法提供实时的多用户协和操作，需要颠覆式地构思新的交互界面。

PyMT 运行时以树状目录组织 widget. 根目录是应用程序窗口(一般是 MTWindow). Widget 可以通过 MTwidget 基类的 add_widget 方法加进这样的目录中或者加进其他 widget 中.各种事件如 on_draw, on_resize, mouse event 和 touch event 可以通过这个树状目录到达所有的 widget. 程序员可以利用这些层级机构来定义各种容器(container), 用以处理布局和控制 widget 对事件的响应.

PyMT 提供了丰富的功能强大的 widget 和实用的对象(object). 比如,所有的 widget 可以用 CSS 来定义风格. 除了用 add_widget 来生成 widget 之外,还可以用 XML 来定义层级结构,然后自动生成 widget. 由于 PyMT 的功能实在太多,无法一一介绍,详情请参阅 PyMT API 文档.

接下来的小例子将尝试展示 PyMT 的关键概念. 图 2 展示了该例子的最终效果,用一只手获取 5 个输入点. 这段代码可以分为 4 个部分. 评价一个 NUI 交互系统的唯一也是最有效的方法就是亲自去做,去体验.

1. 导入 PyMT 并初始化参数：第一行代码告诉 python，我们要用 PyMT，这行代码将会加载所有的 PyMT 对象和函数。这段代码同时还设置了一个名叫 touch_positions 的变量。这个变量用以存储触摸事件的坐标位置。
2. 定义一个新的类，名叫 TestWidget：这个类继承自 MTWidget 并定义了 4 个事件处理器。on_touch_down 和 on_touch_up 事件处理器更新触摸位置。on_touch_up 处理器从触摸列表中删除触摸。draw 方法会在每个触摸事件的

当前位置生成一个半径为 40 的圆。这个圆通过 PyMT 的 drawCircle 方法，并调用 touch_positions 的值生成。

3. 创建一个窗口来装载 widget：MTWindow 是一个应用程序窗口，你可以通过 add_widget 方法来加载 widget。被加载的 widget 会通过窗口接收 touch 和 draw 事件，并渲染该窗口。
4. 启动程序：runTouchApp 函数会启动 PyMT 的主程序，同时任命窗口，打开 TUIO 侦听器，并开始发送事件。

```
from pymt import *
#a dictionary/hash map to store touch positions
touch_positions = {}
#A widget that will draw a circle for each touch
class TestWidget(MTWidget):
#these functions handle the touch events
def on_touch_down(self, touches, touchID, x, y):
touch_positions[touchID] = (x,y)
def on_touch_move(self, touches, touchID, x, y):
touch_positions[touchID] = (x,y)
def on_touch_up(self, touches, touchID, x, y):
del touch_positions[touchID]
#at each frame, the draw method draws the widget
def draw(self):
set_color(1,0,0)
for position in touch_positions.values():
drawCircle(position, radius = 40)
#create a window and add our widget to it
win = MTWindow()
widget = TestWidget()
win.add_widget(widget)
#run the program
runTouchApp()
```

程序 1：PyMT 程序示例，在每个触点处生成一个红色的圆。

图 6：程序 1 所示程序运行截图。5 个触点（屏幕分辨率：640x480)。

2.4.7 多点触摸输入编程

基于多点触摸设备的编程和交互设计与基于鼠标的界面的情况完全不同。前面章节讨论了 NUI（自然交互界面）与传统的基于 GUI（图形界面）的 WIMP 的区别。主要的区别在于多点触摸给交互界面带来的无限的拓展和可能性，同时也让编程变得更加复杂。

TUIO 方式，以及其他多点触摸协议和框架定义了 3 种基本的触摸消息/事件。包括新的触摸事件，现有触摸事件的移动，触摸事件的移除。这些消息总是被标明“touchID”，这样程序以及这些事件调用函数就能把这些触摸事件各自区分开来。如程序 1 中所示，这些触摸事件以下面所示的方式到达 PyMT：

- on_touch_down(touches, touchID, x, y)

- on_touch_move(touches, touchID , x, y)
- on_touch_up(touches, touchID, x, y)

每个事件都携带着 touchID 和 x，y 坐标值。通过这些 touchID 和坐标值，系统就区分目前所有的触点及其位置。实际上，系统不仅能区分目前所有触点的位置,还记录着目前每个触点的移动和加速度,这些都由 TUIO 协议定义。PyMT 也为 TUIO 对象提供 on_object_*事件（比如：图形标签的识别)。

很明显，解释多点触摸要比处理单一指点设备如鼠标复杂得多。任何一个触点都会影响到用户界面,随后的事件处理器在决定如何处理触摸事件之前必须兼顾所有其他可能发生的相互作用。

在进行 PyMT 编程的时候，一些基本的编程技巧已经被证实很有帮助。比如，让某个 widget 对某个特定的触摸拥有所有权在很多案例中被证明很有效。当使用这个技巧的时候，其他的 widget 会忽略某个特定 touch_ID 所对应的 touch_move 或者 touch_up 事件。只有对这个事件拥有所有权的 widget 才会处理这些事件。

基本上，多点触摸和基于多点触摸的交互界面潜力无限，如果一定要有什么限制的话，那么，这些限制就是我们的创造力和想象力。多点触摸的手势有无穷的组合方式。在接下来的例子里，你将看到，这些手势及其组合可以形成很多直观的交互操作，但是，这也意味着逻辑和算法变得更加复杂，更加困难。

2.4.8 例子：实现旋转/缩放/移动

这一部分，我们将讨论如何实现一个著名的操作。旋转/缩放/移动（图 7）是最常见的多点触摸演示示例。这种直观的，用两个手指旋转/缩放/移动一个二维物体的方式，和我们在桌面上移动一张纸的情况很相似。我对这个方式感觉很

直观，很自然，因为两个触点始终在物体上一开始的那个位置，无论手指移动到哪里。

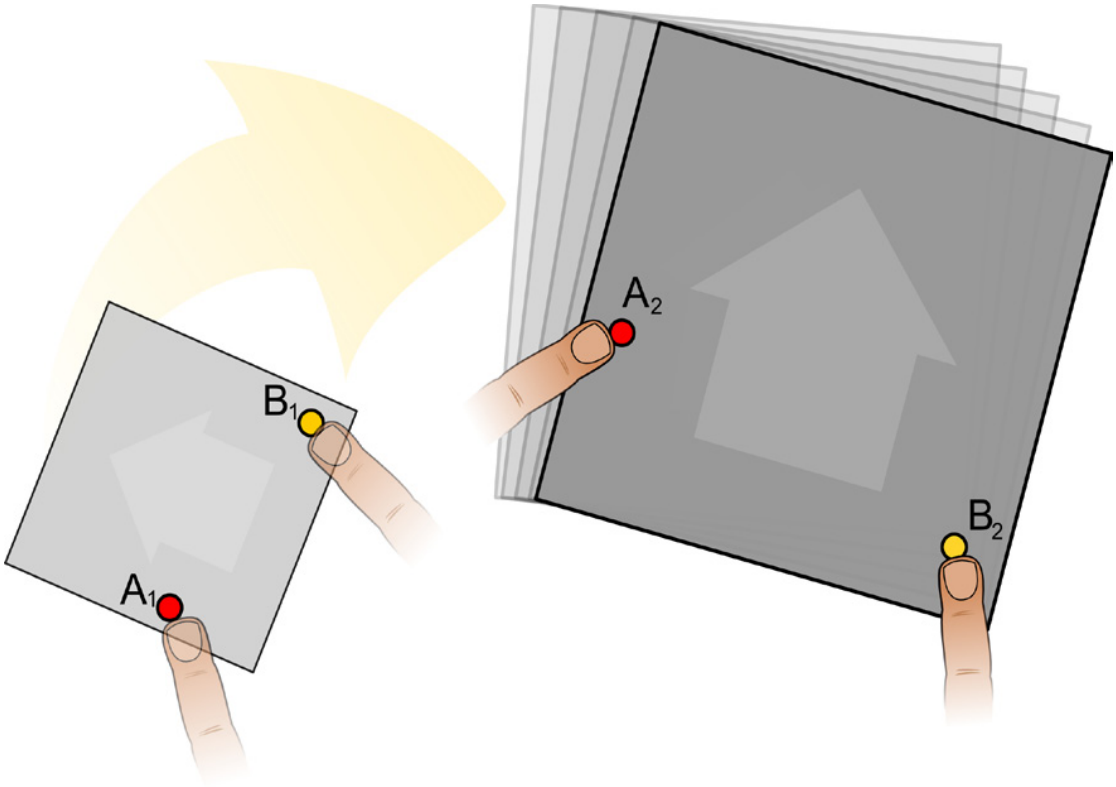


图 7：旋转/缩放/移动。用户可以用两个手指触摸物体，并把它移动到其他地方。

在 PyMT 中，这个交互方式以 ScatterWidget 对象实现，你可以在这个对象中添加其他 widget，让他们成为 scatter widget 的一部分。诚然，这个交互方式已经用了太多次了，有人或许会说这个方式已经用烂了。我们在这里讨论这种交互方式不是为了炫耀，而是因为它是一个非常不错的例子，一个典型的证明多点触摸编程复杂性的例子。尽管目前只用了两个触点，而整个交互非常自然，但是这里面涉及到的计算和数学知识已经比鼠标交互复杂了很多。

为了理解这个交互的具体实现，需要对矩阵变换有基本的了解。通过矩阵乘法，任何变形都可以通过一个矩阵实现（一般来说是 4x4 矩阵）。比如一个矩阵

表示沿 x 轴移动 5 个单位，可以乘上一个绕 y 轴旋转 90 度的矩阵。得到的矩阵则表示在沿 x 周移动 5 个单位的同时，绕 y 轴旋转 90 度。更多矩阵相关信息，参见[18]。

2.4.9 用矩阵变换来画图

程序 2a 是做旋转/缩放/移动的第一部分。这一部分用于生成对象。Transform_mat 是一个变形矩阵。到目前为止它只用来保存标准矩阵，让所有的点保持不变。基于触点的位置和移动，对变形矩阵进行修改，达到改变对象的目的。

```
#a matrix that holds the transformation we are apply-ing to our object
self.transform_mat = IdentityMatrix()
#a hashmap to keep track of the touchID's we have seen
self.touches = {}
#this function draws the object using the transfor-mation matrix
def draw(self):
    glPushMatrix() #save the current matrix state
    glMultMatrixf(self.transform_mat) #apply transfor
        self.draw_object() # draw the object as usual
    glPopMatrix() #restore the current matrix state
```

程序 2a.旋转/缩放/移动示例。当生成对象的时候，应用了一个变形矩阵，这个矩阵会随着触点位置的变化发生改变。详细的代码参见 pymt/ui/scatter.py 参考文献[2]

2.4.10 确定参数并计算变形

接下来的问题就是决定如何进行变换对象的变形矩阵。图 8 描述了需要变换的部分参数。重要的一点是，对任何一个给定的事件，两个触点中，只有其中一个可以移动。因为，每个事件一次只能传送一个触点的信息。

为了计算变换，需要以下参数。事件发生前和事件发生后两个触点的距离（d1 和 d2）。角度 R，用来计量对象的旋转角度。旋转和缩放中心点 Pi（两个触点的其中一个）

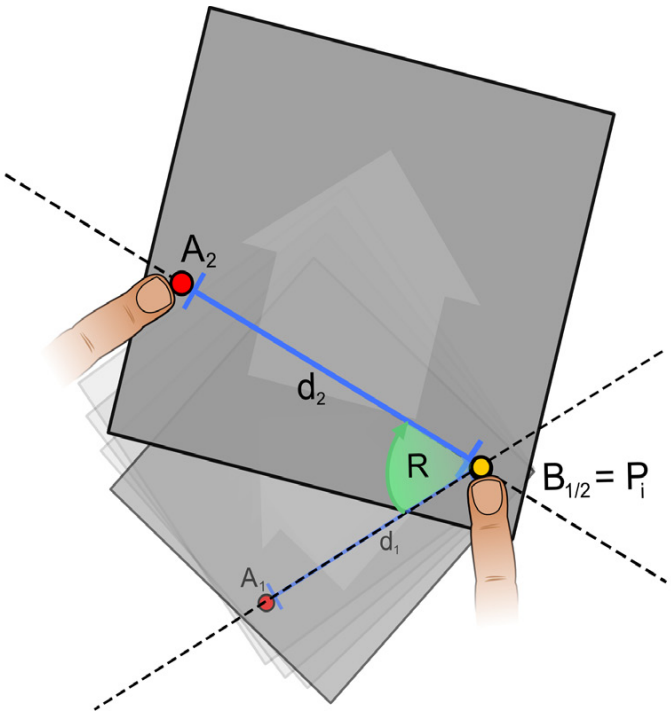


图 8 旋转/缩放/移动。两个触点（A 和 B），对多个参数进行计算。缩放值（d2/d1），旋转角度(R), 旋转和缩放中心(Pi).每次只有一个触点位置发生改变（A=Pi 或者 B=Pi）

通过对参数的计算，变形矩阵会做出相应的修改。程序 2 展示了 PyMT/OpenGL 是如何完成这些的。具体的步骤如下：

1. 初始矩阵被装载进 transform_mat
2. 通过变换，让旋转中心的坐标为（0，0），因为，在 OpenGL 中，旋转和缩放总是围绕原点进行。
3. 绕 z 轴旋转（z 轴垂直屏幕）
4. 缩放
5. 把所有的物体移动到当初的位置

另外一件值得注意的事是，对象没有发生移动（平移）。运用这种技术，物体的运动通过绕固定的点旋转或者在不同方向上缩放实现。如果要移动对象的

话，比如用一个手指拖动，那么这个拖动的动作必须添加到 PyMT 的 ScatterWidget 类中去。为了保持例子的简洁，这里就不深入探讨。

```
#this functions applies the newly computed transformations
#to the old matrix
    def apply_angle_scale_trans(self, angle, scale, point):
        glPushMatrix() #save the current matrix state
        glLoadIdentity()
        glTranslated(point.x, point.y,0) #5. move back to intersection
        glScaled(scale, scale,1) #4. Scale around (0,0)
        glRotated(angle,0,0,1) #3. Rotate around (0,0)
        glTranslated(-point.x, -point.y,0) #2. Move intersection to (0,0)
        glMultMatrixf(self.transform_mat) #1. apply transform as was
        #now save the modified matrix back to self.transform_mat
        glGetFloatv(GL_MODELVIEW_MATRIX,self.transform_mat)
        glPopMatrix() #restore the current matrix state
```

程序 2b。应用变形参数示例。这个函数把变形参数应用到变形矩阵上。完整的程序见 pymt/ui/scatter.py 参考文献[2].

2.4.11 触摸位置转译成计算参数

最终，程序必须根据触摸事件提供的位置单独计算参数。程序 2c 展示了相应代码。为了简洁，这段代码假设 get_second_touch 函数用于获取第二个触点的信息。这样的程序可以通过一个库来跟踪记录 touchID，确保每次只有两个点在这个库中，然后返回与被传递的参数不同的那个。

这段代码也假设了一些基本的矢量函数，用于计算两点间的距离和角度。具体的实现参见 PyMT 的 vector 类[2]。或者参考[20][21]。这个计算基本上只是两个向量的大小和乘积。

旋转的角度通过 A1-B 和 A2-B 得出。缩放的比例通过 d2/d1 得出。

```
def rotate_zoom_move(self, touchID, x, y):
    intersect = Vector(0,0)
    rotation = 0
    scale = 1
    #we definitly have one point, so if nothing else we drag/translate
```

```
    A1= Vector(*self.touches[touchID])
    A2= Vector(x,y)
    #get the second touch
    #(not the on with touchID of current event)
    second_touch = self.get_second_touch(touchID)
    B = Vector(*self.touches[second_touch])
         #compute scale factor (d1 and d2 are floats)
    d1= A1.distance(B)
    d2= A2.distance(B)
    scale = d1/d2
    #compute rotation angle
    old_line = A1 - B
    new_line = A2 - B
    rotation = -1.0 * old_line.angle(new_line)
    #apply to our transformation matrix
    self.apply_angle_scale_trans(rotation, scale, B)
    #save new position of the current touch
    self.touches[touchID] = Vector(x,y))
```

程序 2c。计算旋转/缩放/移动参数的程序示例。详细代码见 pymt/ui/scatter.py 附录[2]。

2.5 ActionScript 3&Flash

时间回到大约 1996 年，当时网络还相对来说比较年轻，人们在寻找一种工具以帮助其网站富有生机和鲜明醒目。在这时，FutureSplash 出现了，也就是 Flash 的前身，通过在网站植入 Macromedia 的 Shockwave 播放器的方式来绘制简单的矢量图形动画。Macromedia 公司在不久后就收购了 FutureSplash Animator 并重新命名为 Macromedia Flash。这些年来，Flash 不断演化，能力增强到遍布全世界的电脑。在 2000 年，Macromedia 增加了 ActionScript 1.0，用编程语言的方法来编辑和操作对象，而不是仅仅用时间轴绘画。在 2005 年，ActionScript 2.0 开始将面向对象编程思想融入到语言当中，而不是让 ActionScript 仅有一个简单的语法。在 2007 年，Adobe 收购了 Macromedia，新版 Flash 推出，命名为 Adobe Flash CS3。Adobe 对 ActionScript 版本进行了全面的修订和增改，推出 ActionScript 3.0。

到现在，Flash 已经是网页设计师最强大的工具之一，然而，Flash 已经不仅仅局限于网络。由于名为 Adobe Flex 和 Adobe AIR 的新的编程平台的出现，开发者可以用 ActionScript 3.0 去开发跨平台(Cross-Platform)的桌面应用程序。现在，Flash 可以来创作多点触摸应用，通过计算机视觉技术和多点触摸感测程序，比如 Touchlib，CCV 和 reacTIVision 的协助。

2.5.1 理解通信过程

在我们开始用 Flash 开发多点触摸程序之前，理解多点触摸感测程序的信息传递流程是很重要的。当你启动一个已经准备好的程序开始发送触点信息时，程序可以通过 TUIO 协议来读取它。然而，Flash 并不能马上理解这些触点信息。这是为什么呢？“TUIO 是一个建立在 UDP 通信协议上的一个非常简单的协议，所以当你希望使用 Adobe Flash CS3 或者 ActionScript 3.0 来创建一个能快速响应的程序时，你需要有一个桥梁来读取 UDP 接口信息然后转变为 TCP 连接模式”

什么可以让 UDP 转变为 TCP 连接模式呢？Flash OSC 就可以。[3]。它可以在我们需要的 Touchlib 和 Flash 之间搭一座桥，让你可以创建 Flash 多点触摸应用。

Inside MyFirstApp.as，粘贴这些代码：

```
package app.demo.MyTouchApp {
    import flash.display.*;
    public class MyFirstApp extends Sprite {
         public function MyFirstApp():void {
               trace(“MyFirstApp Loaded”);
               }
         }
   }
```

在调整你的 FLASH 文件属性后，需要再加一点工作。找到你的 Document Class(文档类)属性，并且填入 app.demo.MyTouchApp.MyFirstApp，这个文件就和 MyFirstApp.as 文件连接起来了。

还有最后一件事需要做，现在的 Flash 文件是空的，并且要让 TUIO 工作的话，需要有一些东西在舞台上以让它识别出用户正在触摸屏幕。现在，我们放一个 Shape 覆盖舞台，使用矩形工具并让它填满整个舞台。然后设定颜色，在 Flash

CS3 中，你甚至可以将 Alpha 值调制完全透明。TUIO 就可以识别 shape 并且做出响应。

现在运行“控制”菜单里的“运行”按钮，测试影片。

现在，我们添加一个 TUIO 类到这个.as 文件中。

```
...
        public function MyFirstApp():void {
        TUIO.init(this,'localhost',3000,'',true);
        }
        }
}
```

再次测试影片，你应该看到白、红、绿的正方形在左上角，触摸的信息在右上角，而且当你在舞台上触摸的时候，在你的手指下面出现圈圈。

好了，一切准备就绪。现在，我们需要解决一下如何来把收集的触摸信息添加到一个数组里。我们创建一个 BlobLines.as 文件，并在里面添加一个数组，每次触摸按下的时候都会将这些触点信息添加到这个数组里，而触摸抬起的时候又把触点信息从数组里移除。当做拖拽触摸的时候就更新数组里的所有信息，我们可以自己写这样的数组，但那实在是挺烦人的。但是！我很高兴在 TUIO.as 里的数组 OBJECT_ARRAY 已经包含了这些信息，它要比我们自己写这个 AS 文件更简单。TUIO.as 直接从 FLOSC 收集信息，所以看起来要更可靠和准确。

那么我们该如何做呢？我们只需要在 flash/event/TUIO.as 里添加一点东西即可。

```
public static function returnBlobs():Array {
        return OBJECT_ARRAY;
}
```

OBJECT_ARRAY 是一个私有变量，这个变量的创建是为了能够改变这个数组的值。这是有道理的，因为你不想在当你在使用特定的多点触摸对象的时候

随意地改变变量值。相反，我们不想改变它，只是想在任何时候都能得到它的值，所以在第 119 行左右，加上这个 returnBlobs()函数。

保存文件并回到 MyFirstApp.as 文件。

现在我们测试这些信息，当然我们希望它能通过。

那么在这个过程中，都有些什么东西被添加或者修改呢？

addEventListener(Event.ENTER_FRAME,test_returnBlobs);被添加到一个函数，而且会在每一帧都会运行它。导入 Event 包: import flash.events.Event; 让 AS 文件知道 Event.ENTER_FRAME 是什么，然后用来追踪 TUIO.returnBlobs()。

测试影片。

你应该可以在每帧绘制的时候在文档的属性面板里看到触点的数量。这是好事，我们以后可以在需要的时候直接开始使用。

```
package app.demo.MyTouchApp{
        import flash.display.Sprite;
        import flash.events.TUIO;
        import flash.events.Event;
        public class MyFirstApp extends Sprite {
            public function MyFirstApp():void {
            TUIO.init(this,'localhost',3000,'',true);
            addEventListener(Event.ENTER_FRAME, test_returnBlobs);
        }
            public function test_returnBlobs(e:Event):void {
                trace(TUIO.returnBlobs().length);
            }
        }
}
```

2.6 .NET/C#

根据维基百科(Wikipedia)的解释,Microsoft .NET Framework是由微软开发,一个致力于敏捷软件开发（Agile software development)、快速应用开发（Rapid application development)、跨平台和网络透明化的软件开发平台。.NET Framework 是以一种采用系统虚拟机运行的编程平台，以通用语言运行时（Common Language Runtime)为基础,支持多种语言(C#、VB.NET、C++、Python等）的开发。.NET 也为编程界面（API）提供了新功能和开发工具。这些革新使得程序设计员可以同时进行Windows应用软件和网络应用软件以及元件和服务（web service）的开发。.NET 提供了一个新的反射性的且面向对象程序设计编程界面。.NET 设计得足够通用化从而使许多不同高级语言都得以被汇集。

2.6.1 使用.NET的优势

使用.NET 框架(.NET Framework)最显著的优势在于当你使用这个框架编写程序的时候，可以保证代码能够运行在所有安装了.NET Framework的虚拟机上。.NET Framework已经被预装在微软已发布的最新操作系统(Windows Vista)里了，当然也就包括即将发布的Windows 7。

另一个优点就是代码运行时“自动管理”，意思就是它不会让你的系统崩溃,也不会降低系统的稳定性。从大范围看,兼容性问题也要比原生的C++程序要少。

2.6.2 .NET的历史和多点触摸

.NET 2.0没有一个真正的预期想象的多点触摸典型案例，因为.NET 2.0缺少自由的用户界面。它的目的是作为一个商业的应用程序框架,而不是更加丰富的用户界面。只有在.NET 3，WPF和Silverlight中才提供了被用作多点触摸应用程序开发的足够条件。XAML标记语言是可扩展的和相对自由的，它允许开

发者开发丰富美观且具有交互性的界面。

.NET 3 一开始也不是一个支持多点触摸的平台。Flash 在当时仍然是爱好者们最主要的设计和开发平台，原因就在通过 Touchlib 和 TUIO 已经发展了很多的项目，而且这个方案可以很简单地将触摸信息发送到应用程序上。

在 2007 年，Donovan Solms 创造了 C# Touchlib Interface (CsTI). 它可以通过二进制连接将 Touchlib 获得的触摸数据发送到.NET。CsTI 将触摸事件转换为.NET 经常使用的实际.NET 事件(actual .NET events)。另一种比较常见的方式是使用与 Flash 相同的 TUIO，然后获取触摸数据输入到.NET 中。

从那时起，许多的.NET 多点触摸框架(.NET multitouch Frameworks)被创造出来。使用 MultiTouch Vista，你现在可以通过 CCV 或者 Touchlib 的安装程序来控制 Windows 7。Microsoft Surface 使用了.NET 来作为它的应用程序的基础，.NET3，WPF，Silverlight 都支持 3D。

XNA，微软新的管理图形的 API，拥有更好的 3D 支持，但是目前对它的了解有待继续探索。

2.6.3 开始使用.NET 来开发多点触摸应用

首先，您需要决定是使用现有的框架还是自己开发新的框架，或者根据您的需求来扩展现有框架，毕竟它们很多都是开源的。下面是 3 种备选方案以及如何使用它们进行开发。

使用现有的框架

现在确实有不少的.NET 多点触摸框架(.NET multitouch Frameworks)可供选择，它们一般被命名为 WPF 多点触摸框架。比如 MultiTouchVista 支持 Windows 7 上的多点触摸交互。

自己开发新的框架

这个方案适用于有经验的开发者。它需要与原始触摸数据工作，所以要为这个框架解决如何建立一个事件系统，以及之间的算法，确定所有的底层操作和.NET 一般性的处理架构。

这儿有两种途径来获取原始的触摸数据然后输入到.NET 中：

C# Touchlib Interface (CsTI) 和 通过 XMLSocket 连接到 TUIO

其中，CsTI 是一个只能通过 Touchlib 来工作的二进制连接方案，而通过 XMLSocket 连接的 TUIO 可以与 Touchlib，CCV，ReacTIVision 中的任何一个进行协同工作，你可以从 ReacTIVison 网站上获得基本的实现过程的演示(http://re-activision.sourceforge.net/)。现在，第一个.NET 多点触摸框架已经不被社区所支持了，但是它的代码仍然可以在 Google Code 上找到，来作为学习的起点，只是起点而已。现在更多的人使用的是 IInputProvider。关于这个，你可以在 MSDN 上找到相关的解释。

扩展现有的框架

这种方案适用于开发者找到了一个比较合适的框架，但是里面缺了自己需要的个别功能。在这种情况下，你只需要购买(有的框架是收费的)或免费获取你所需要的部分，而后在它的许可证下进行二次开发，注意版权问题，哪怕是开源的。还有注意它的工作方式，以及这个框架现在是否还很活跃。

工具

大多数的.NET 程序员更喜欢微软的 Visual Studio，这个确实是一个完善的，多功能的.NET 集成开发平台(IDE)。你可以从微软的相关页面上在线安装或者下载完整的离线安装包(ISO 格式)，Express Editions(速成版)是免费的。

2.7 框架和类库(Framework and Libraries)

英文注释说明

Programming Language *编程语言*

License *许可证*

Page *项目网站页面*

2.7.1 计算机视觉(Computer Vision)

BBTouch

BBTouch 是一个开源的,运行在 OS X 上的跟踪视觉的多点触摸模拟运算表(MultiTouch tables)环境

Programming Language: Cocoa (Mac)

License: GPL license

Page: http://benbritten.com/blog/bbtouch-quick-start/

Bespoke Multi-Touch Framework

Bespoke Multi-Touch Framework 是一个功能丰富、可扩展的多点触摸开发框架。在 BSD 许可证下开源发布，你可以自己使用并扩展源代码以满足你的需求。

该框架可搭配任何基于视觉的多点触摸硬件平台[比如红外光谱(FTIR)或扩散照明(Diffused Illumination)]在这个包里包括一些示例程序，一个 Windows 鼠标模拟器，2D 符号识别器，4 点校准器和一个独立的表现层(Presentation Layer,支持 XNA 和 WinForms)，OSC 网络支持单播，多播和并发广播的 UDP/IP 协议。

Programming Language: C#

License: BSD License

Page: http://www.bespokesoftware.org/multi-touch/

reacTIVision

reacTIVision 是一个开源的、跨平台的、强大的计算机视觉框架，能够快速地识别附加到物理对象上的附加标记，也可以用于多点触摸的识别。它的主要目的是作为一个工具包用来快速构建表格基准(table-based)的有形的用户界面[tangible user interfaces (TUI)]。

Programming Language: C++

License: GPL license

Page: http://reactivision.sourceforge.net/

Community Core Vision (CCV)

Community Core Vision，缩写为 CCV，之前称作 tBeta，是一个开源的、跨平台的计算机视觉与多点触摸感测解决方案。它需要一个视频输入流和跟踪数据输出(比如坐标系和触点大小)以及在多点触摸应用中的触摸事件(比如手指按下，移动和释放)。CCV 可以连接多种网络摄像头和视频设备以及连接到各种启用 TUIO/OSC 的应用程序上，支持多种多点触摸光线处理技术，包括 FTIR，DI，DSI，LLP 以及未来扩展的计划(自定义模块和筛选程序)。

Programming Language: C++

License: MPL or MIT (not defined)

Page: http://tbeta.nuigroup.com

Touché

Touché是一个免费且开源，用来追踪光线的多点触摸模拟运算表。它已经写入了 Mac OS X Leopard 并使用了它的许多核心技术，如 QuickTime Core Animation、Core Image 和 Accelerate 框架，也包括了像 libdc 1394 和 OpenCV 这样的高品质开源库，然后去实现良好的追踪性能。

Programming Language: Cocoa (Mac)

License: LGPLv3

Page: http://gkaindl.com/software/touche http://code.google.com/p/touche/

Touchlib

Touchlib 是一个建立多点触摸交互界面的类库。它处理红外光并追踪触点，然后向您的程序发送多点触摸事件，如“手指按下”、“手指移动”和“手指释放”等。它包括一个配置程序和几个用于入门的演示，以及与大多数类型的网络摄像头和视频采集设备兼容并互动。目前只能运行在 Windows 上，但是开发人员正努力把它移植到其它平台。

Programming Language: C++

License: New BSD License

Page: http://nuigroup.com/touchlib/ http://code.google.com/p/touchlib/

2.7.2 网关程序

FLOSC

FLOSC 是一个通过“FLOSC Server”通信的 AS3 类库，它能够使 Flash 程序获得 OSC 信息。

Programming Language: Java

License: MIT

Page: http://code.google.com/p/flosc/

2.7.3 客户端

Creative multi-touching

Creative Multitouching 是一个运行在多点触摸环境下的工具，其目的是为了能够促使多点触摸环境下的创新项目的诞生。比如绘画、简单的书写以及从 Flickr 和 YouTube 上寻找图片和视频并将它们组合成创意拼贴。

Programming Language: Actionscript 3 (Adobe Air)

Status: active

License: not specified

Page: http://www.multitouching.nl/page.asp?page=148

Grafiti

一个致力于互动桌面界面的跨平台、可扩展的手势识别管理框架。它建立在 TUIO 客户端上，支持 MultiTouch 手势界面的开发，包括使用有形对象作为追踪目标[译者注：有形对象指的是比如在上面放上个方块或者别的能看得见、有体积的东西]。

Programming Language: C#

License: GNU General Public License (GPL) v3

Page: http://code.google.com/p/grafiti

Multi-Touch Vista

Multi-Touch Vista 是一个能够处理多种输入设备(Touchlib、多键鼠标、Wii 的遥控器等)的用户输入管理层，能够使当前窗口得以缩放和旋转。它允许标准的应用程序使用多点式的缩放和旋转。它还为多输入的 WPF 程序开发提供了一

个框架。MultiTouch Vista 支持 Windows XP/Vista。

Programming Language: C#

License: GNU General Public License (GPL) v2

Page: http://www.codeplex.com/MultiTouchVista

PyMT

PyMT 是一个以 pyglet 为基础的多点触摸富媒体 OpenGL 程序的 Python 开发模块。它的目的是可以快速、易于交互设计和快速原型开发。还有一个侧重点就是能够对用户互动数据进行量化分析和可视化处理之后存储。

Programming Language: Python

License: GPL v3

Page: http://pymt.txzone.net/

TouchPy

TouchPy 是一个纯粹的轻量级 Python 多点触摸框架，它不限制你使用任何的 GUI Toolkit。使用简单，所以也是最通用的 Python 多点触摸框架。

Programming Language: Python

License: GPL

Software & Applications 63

Page: http://touchpy.googlecode.com

2DCur

一个能够控制从 OSC/TUIO 协议 2DCur(2D cursor)信息事件的项目。它是一个外部的，在 Firefox 可视化编程环境下的 Python 框架。

Programming Language: Python, Lily (Javascript Visual Language on Mozilla

Framework)

License: GPL3

Page: http://2dcur.googlecode.com

2.7.4 模拟器

SimTouch

SimTouch 是另一个使用 Adobe AIR 运行时的 TUIO 模拟器。最大的优势在于透明的背景能够使开发者更好的了解触摸的事件。

Programming Language: Action Script 3 (Adobe Air)

License: MIT License

Page: http://code.google.com/p/simtouch/

ReacTIVision

reacTIVision 是一个开源的、跨平台的计算机视觉框架。能够快速稳健地追踪在有形物体上的基准标记或者是手指的多点触摸。它的主要目的是作为一个工具包，以便快速开发以模拟运算表为基础(table-based)的有形用户界面(tangible user interfaces，TUI)和多点触摸互动表面。这个框架是由 Martin Kaltenbrunner 和 Ross Bencina 作为 reacTable 项目在西班牙巴塞罗那 Universitat Pompeu Fabra 的 Music Technology Group 开发的，它是一种新型的桌面多点触摸界面的电子音乐文书。

Programming Language: Java

License: GNU General Public License

Page: http://mtg.upf.es/reactable/?software

QMTSim

这个项目的目的是建立一个新的快速的多点触摸程序 TUIO 开发调试模拟器。TUIO 是一种多用途协议，尤其适用于桌上有形用户界面，而且突破了人们开发调试多点触摸应用程序必须得有多点触摸硬件的限制。

Programming Language: c++

License: GNU General Public

附录 A：词汇解释

多点触摸（Multi-touch）：一种允许单用户或多用户同时进行操作的图形交互技术。

多点（Multi-point）：一种利用各个点之间的关系，而不是其运动进行操作的交互技术。比如：一个支持多点触摸的广告亭。

多用户（Multi-user）：支持多个用户使用的多点触摸设备。大型的多点触摸设备很自然的让人联想到多用户操作。

多点模式（Multi-modal）：一种支持多模式界面系统的交互形式。

笔记本电脑计算（Tabletop Computing）：显示屏支持交互的笔记本电脑。

直接操作（Direct manipulation）：通过人的肢体（手，手指等）直接操作数字化界面。

触点追踪（Blob tracking）：给每个点分配一个 ID 号（识别器）。对每一帧画面进行分析，区别出每个触点，并与其在上一帧的状态进行对比。

触点检测（Blob detection）：从图片中检测出比环境更亮或者更暗的区域的过程。

追踪器（Tracker）：一种程序，该程序能获取摄像头的画面，并通过各种滤镜提取其中的点，然后通过某种协议，报告这些点的位置、尺寸以及各自运动。

TUIO：可触摸的用户界面对象（Tangible User Interface Objects）— 一种用于传送点的位置、尺寸以及相应移动方向等信息的协议。该协议基于 OSC，而 OSC 又基于 UDP。

触摸事件（Touch event）：某个系统感知到有物体接触到该多点触摸设备。

手势（Gesture）：一组可以被感知的物理运动，并且通常会被赋予一定的涵义。比如，一个手指可以进行拖动操作，两个手指则可以进行缩放操作。

传感器（Sensor）：一种计量环境变化的设备。

ZUI：可缩放的用户界面（Zoomable User Interface）— 一种可以无限缩放的用户界面。理论上讲，这种界面会给你一个无限大的工作区域，如果不是因为内存的限制的话。

散射体（Diffuser）：用来散射光线的物体。在很多多点触摸技术中，散射体用来创造一个均匀的照明。

FTIR：受抑全内反射（Frustrated Total Internal Reflection）— 一种利用全内反射特性的多点触摸技术。当有手指或者物体触摸到表面时，会破坏全内反射特性从而产生亮点。

DI：散射照明（Diffused Illumination）— 一种利用散射表面形成阴影（前置式 DI）或者亮点（背投式 DI）的多点触摸技术。有些时候，这种技术和直射式类似。

LLP：激光平面（Laser Light Plane）— 一种利用线性激光发射器生成激光平面的多点触摸技术。当这个激光平面被物体破坏时，相应的区域便被照亮。

DSI：散射式表明照明（Diffused Surface Illumination）— 一种利用特殊的导光亚克力，从侧面打光，生产均匀光照的技术。效果和 DI 类似。

Stereo Vision or Stereocopic：立体视频或立体图像，一种利用两个摄像头的多点触摸技术。

零压力（Zero force）：指触发触摸事件所需的力度或者压力，在这里的“零压力”指“很小的压力”。

附录 B：制作一个 LLP 多点触摸设备

LLP 装置区别于本书介绍的其他装置，其他装置采用红外 LED，而 LLP 采用激光发射器。激光发射器用来在触摸屏表面之上生成红外激光平面，而其他的方案如 FTIR 则是在触摸屏里面打光线，DI 设置则是在触摸屏下方打光。在这一点上，LLP 与 LED-LP 类似。LLP 最大的特点是设置简单，而且触点的对比度非常高。

第一步：所需材料

·透明的触摸屏（亚克力，玻璃，空气）

·红外护目镜

·一字激光发射器（780—880nm）

·投影屏幕或者液晶显示器

·投影仪（如果用液晶显示器则不需要）

·红外摄像头，配上相应波段的红外带通滤镜

如图 1 所示，在 LLP 设置中，红外光线在触摸表面之上，让红外光线尽量贴近触摸表面。当有物体贴近触摸表面，光线便被反射，从而被摄像头捕捉到。

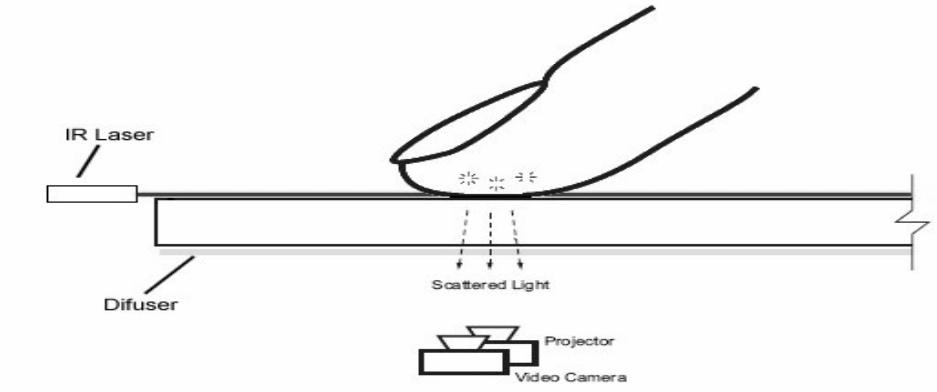


图 1：LLP 原理图

LLP 设置有一个比较大的缺陷，当激光发射器比较少的时候，某些物体会挡住光线，导致放在它后面的物体接受不到光照，导致某些物体无法产生触点。

第二步：安全事项

在我们建造 LLP 设备之前，有些安全事项必须提醒你一下。当你建造 LLP 时，不可避免的要用到红外激光，而这是很危险的。绝对不要把激光对准你自己或他人，即使在有线性透镜的情况下。

图 2：红外护目镜样品

另外，还有两点，确保建造过程中的安全。首先，戴上与你的激光发射器相应波长的红外护目镜（如图 2 所示)。这个原理与带通滤镜相反，正好截止激光发射器发出的光线，而让其他光线通过，确保你的护目镜的波长和激光发射器的波长匹配，要不是起不到任何作用的。

其次，人们常做的一件事就是建一个不反射的“围墙”，就是把触摸区域围起

来的几条边，只有几毫米高。这些“围墙”可以防止红外光线逃逸出来，让它们都呆在触摸区域内。

第三步：激光发射器

很明显，如果采用普通的激光发射器，把它放置在一个玻璃平面表面之上，效果不会很好。这样只能是单点，一个维度，like a fader on an audio interface.出现这个情况有两个原因：第一，离激光发射器最近的物体会挡住所有的光线，从而，这个物体后面的物体都没有可供像下散射的光线（如图 3 所示）。如图中最右部分所示，手指 1 会散射所有的光线，不会有光线到达手指 2.手指 1 会形成一个很亮的点，而手指 2 完全不会被感知。对于这个问题的最直观解决方案就是额外在增加一个激光发射器（如图 4 所示）。即使这样，也只允许两个手指操作，我们基本可以忽略这种解决方案，这不适合我们的要求。

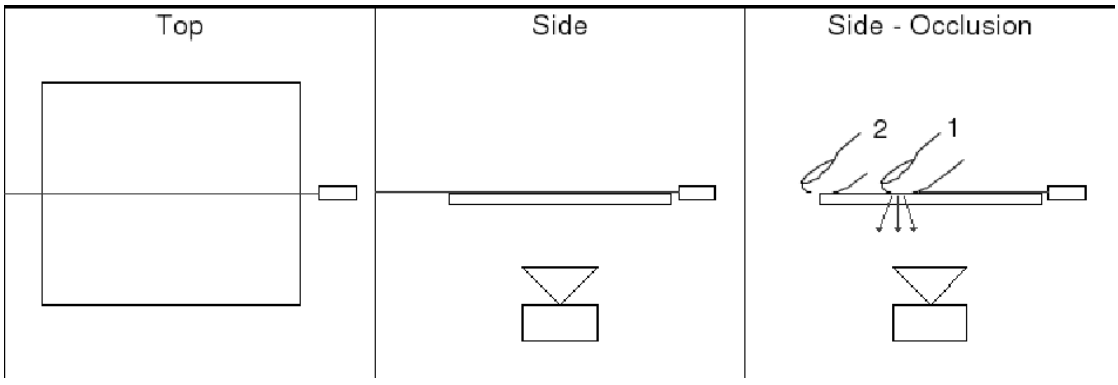


图 3：LLP 光线被挡示意图。

图 3 所示的设置最明显的问题就是激光发射器是一维的。为了解决这个问题，我们需要一个光线平面。为了获得光线平面，我们可以采用一字激光发射器，一字激光发射器就是在普通的激光发射器里加一个线性透镜，在 NUI 里，很多人都使用这种一字激光发射器。

图 3 所示的是采用普通激光发射器的情况。图 4 则展示了使用一字激光发射

器的情况，一字激光发射器会产生一个光线平面，我们在图中看到的只是其中的一个切面。目前的技术情况下，我们无法看到整个光线平面。这种激光发射器就是我们通常在 LLP 多点触摸设备里使用的。

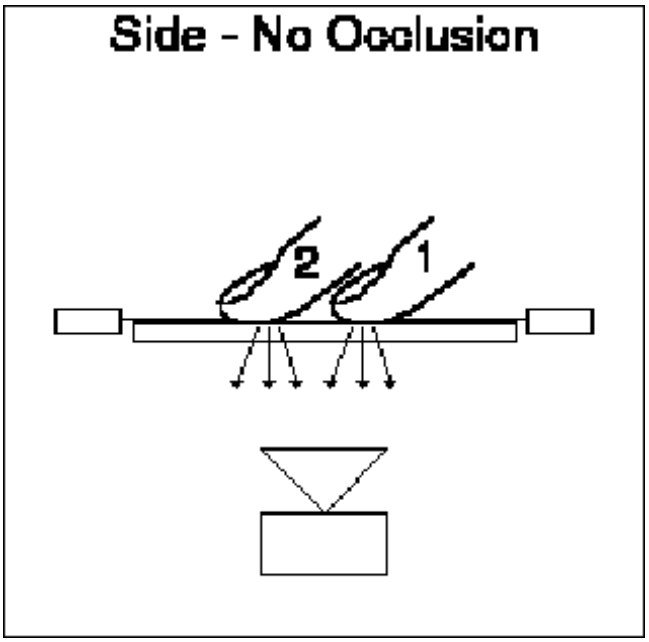


图 4：通过使用多个激光发射器避免光线被挡住

即使我们使用一字激光发射器，同样也会有光线被挡住的情况（如图 5 所示)。图中所示的稍微亮一点的物体会被摄像头看到，而它同时也挡住了照向稍暗一点的那个物体的光线。解决办法就是再增加一个激光发射器。大部分人在 LLP 设置中会用到 4 个一字激光发射器，这样就可以从很大程度上避免光线被前面物体挡住可能性。在使用一个或者两个一字激光发射器的情况下，如果你的设备不是太大或者操作的人数不多于一个，就不用担心光线被挡住的区域会很大。

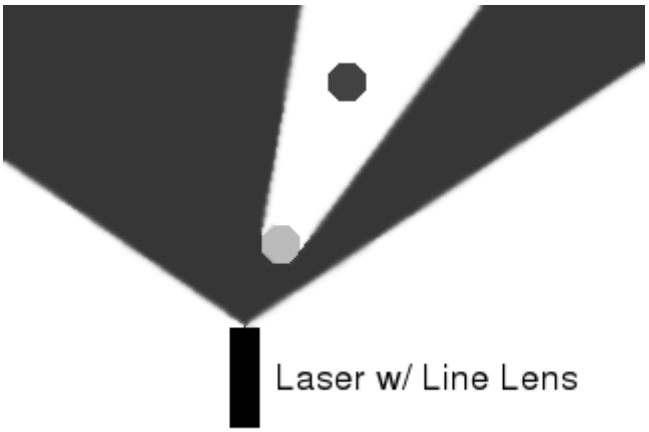


图 5：使用一字激光发射器时光线被挡示意图

第四步：挑选激光发射器

市面上有各种各样的激光发射器，如何找到合适的激光发射器是件很重要的事。和其他元件一样，一分钱一分货。有四个关键因素必须考虑：

·使用几个激光发射器

·功率（以毫瓦 mW 为单位）

·波长

·发射角度

人们通常会去 AixiZ（http://www.aixiz.com）。[中国的用户可以去淘宝上购买，译者注]。首先要考虑的是到底使用几个激光发射器。我们推荐 2-4 个，具体情况根据设备的尺寸来定。激光发射器放置在触摸区域的角落里。如果只使用 1 个激光发射器，有可能会出现光线被挡的情况。理想的情况就是 4 个角落分别放置一个激光发射器。使用激光发射器的原则就是，首先数量要够，然后是质量要好。

下一步工作就是激光发射器的供电。一般的功率都是 5mW，10mW 和

25mW。任何 25mW 及 25mW 以上的激光发射器都更加危险，而且没有必要。更大的功率意味着更亮的光线，同时也更加危险。所以，为了安全，要选择满足需求，但功率小的激光发射器。5mW 和 10mW 的激光发射器基本满足大部分的 LLP 设备。对于尺寸超过 20 英寸（越 51 厘米）的设备，10mW 的效果将会很完美。

此外，还要考虑一下波长，780nm-900nm 都应该很不错。确保你配备了相应波长的红外护目镜。如果你的激光发射器的波长是 780nm（，那么就要准备好 780nm 的护目镜，同时还需要 780nm 的带通滤镜。很多人推荐 850nm 的激光发射器，但这是没有任何依据的。850nm 的并不比 780nm 或者 900nm 效果好。

第五步：选择触摸表面

其实这里没什么要说的，这完全取决于个人喜好。只要保证能够让光线通过就行。

第六步：电源

你需要电源来点亮激光发射器。首要因素是电压。大部分是 3.3V 或 5V。你挑选的电压必须符合你的激光发射器，否则，可能会烧坏它。而给一个 5V 的激光发射器配 3.3V 电源虽然不会烧坏它，但它无法工作。

下一个要考虑的因素是电流。要确保电源能够提供足够的电流，最好还有富余。电流以安（A）或者毫安（mA）计量。粗略的估计一下，每个激光发射器需要 500mA。如果有 4 个激光发射器，则需要 2A 以上的电流。你需要做的就是，启动电源，黑线接地，红线接 5V，或者黄线接 3V。在线路中加一个保险是个不错的想法，另外再给激光发射器加一个指示灯就更好了。很多人用油泥或者腻子固定激光发射器，在这些材料没有硬化前可以调整高度和角度，等他们干了之后，

就成为一个很稳定的支架。

第七步：连接激光发射器

连接激光发射器和连接 LED 很不一样。激光发射器从不串联，也没有必要这么做。我们推荐你进行并联，如图 6 所示。

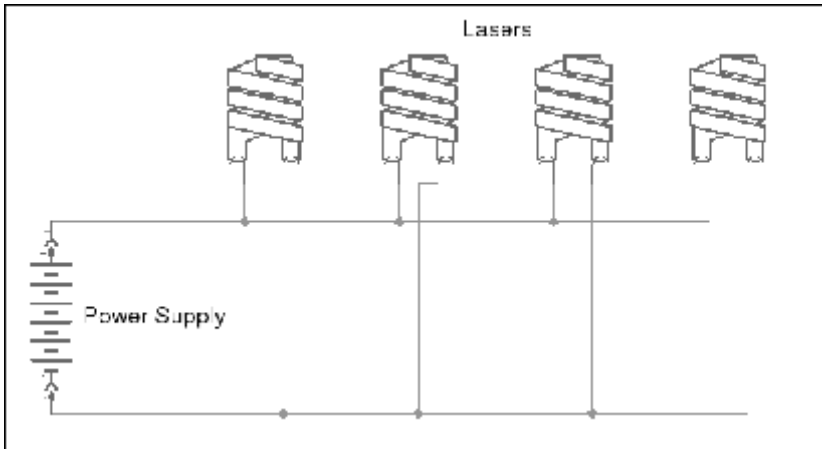


图 6：激光发射器的并联

第八步：装线性透镜并对焦

[使用一字激光发射器可省略这一步，译者注]

这里的介绍可能与从 AixiZ 购买的激光发射器有所不同，但原理上是一样的。激光发射器分为三部分，线性透镜，线性透镜盖，激光发射器本身。如图 7 所示。

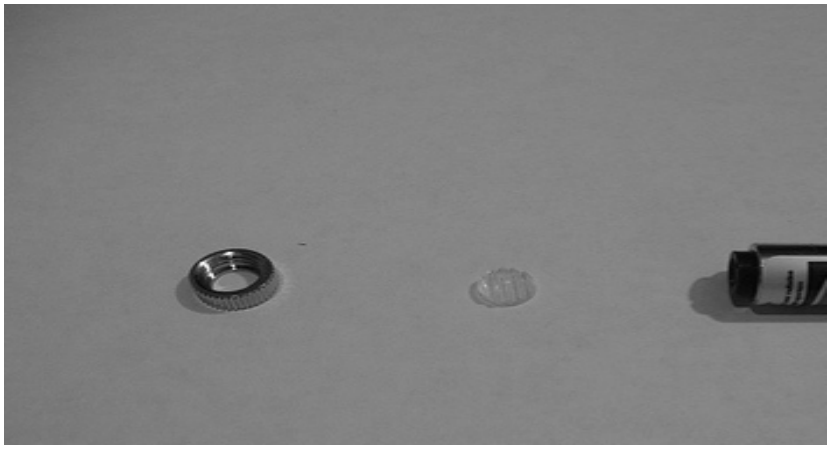


图 7：一字激光器零件（透镜盖，线性透镜，发射器）

把线性透镜放到透镜盖里，有纹路的一面要朝向激光二极管。如果放反了，光线会不直。然后，把黑色部分拧下来，装到透镜盖上，最后再把它们一起装到发射器上。如图 8 所示。

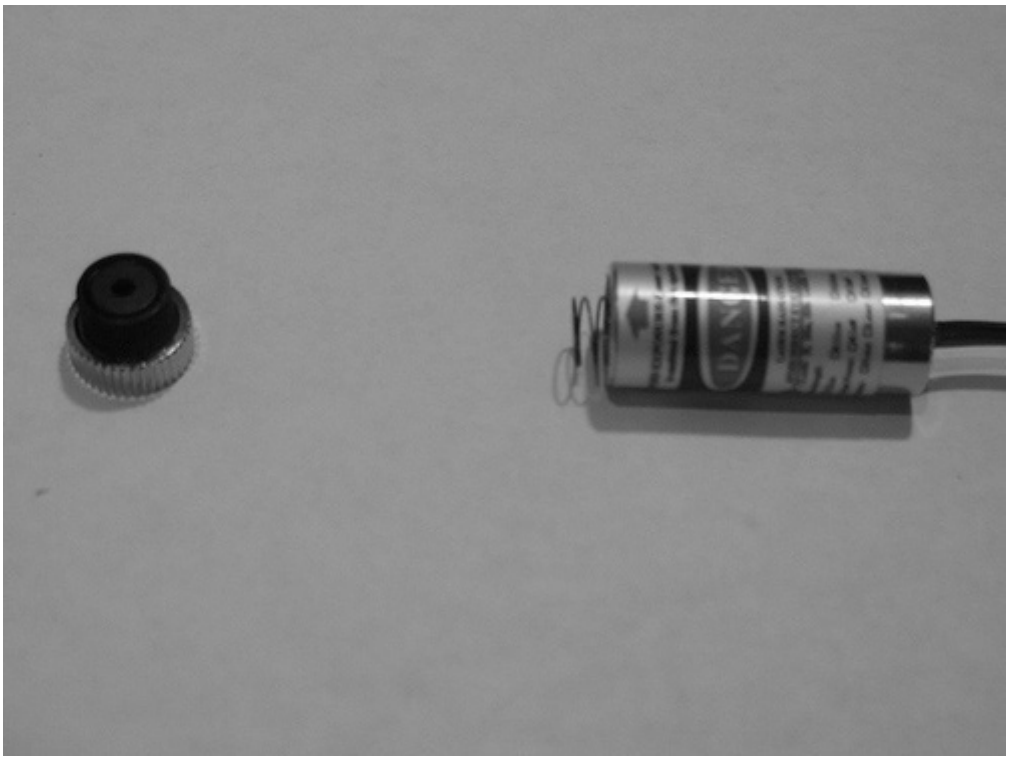


图 8：激光发射器参考组装图

把线性透镜所在的部分拧回一半后（如图 9），戴上护目镜，加上摄像头，点亮激光发射器，对准一张纸。调焦，直到在纸上的线非常清晰。

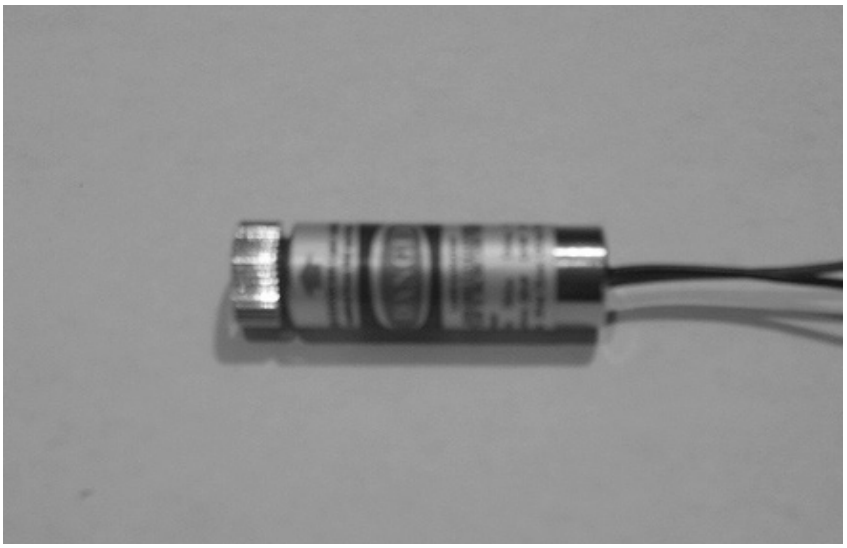


图 9：组装好的发射器

第九步：固定并调节发射器

没有特别好的方法可供推荐,这要根据实际情况来定。人们尝试了各种方法，从胶带到精确到毫米的可调节支架。这和你的设计方案有直接关系，不要受到该教程的限制。

调节发射器是另外一回事。线性透镜所在的部分可以转正负 180 度左右，而不会让光线变得不聚焦，这让调节变得容易。固定好激光头后，拿一张纸对折，形成一条很直的边，可以用来检测激光平面是否和屏幕平行.点亮发射器,用摄像头进行观察，把每个发射器的光线调到和触摸表面平行。要尽可能的让光线贴近触摸表面，避免在物体还没接触触摸表面时便开始反射光线。有很多情况下，可以考虑在发射器后端垫一小片纸，或者你可以找更好的解决方案。

附录 C：制作一个 Rear-DI(背投式 DI)多点触摸设备

红外线从触摸屏下方照亮触摸平面。一个散射体被放置在触摸表面的底部，所以，当有物体接触到触摸表面时，反射的光线比散射体或者背景反射的光线要多。被反射的光线被摄像头捕捉到（使用追踪软件）并被识别为“触点”，同时把这些点的位置转换成 x，y 坐标。图 1 是 Rear-DI 示意图。

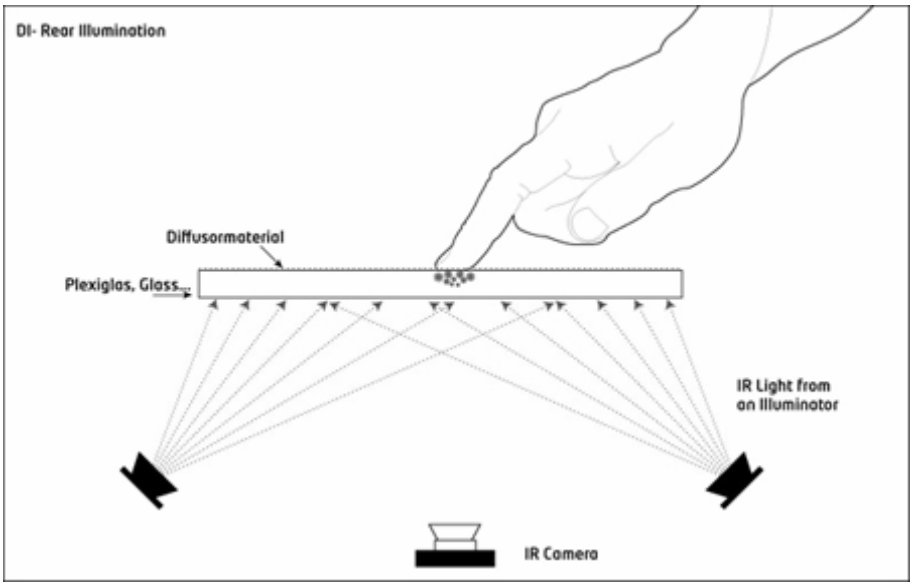


图 1：Rear-DI 示意图

第一步：建一个箱子

做 DI 多点触摸桌的第一步是建了一个封闭的箱子。DI 要求完全封闭的箱子防止红外线溢出。如果有红外线溢出，则会影响到红外线的均匀程度。下图所示的是一个用 Google SketchUp 建立的 ORION 多点触摸桌 3D 模型。

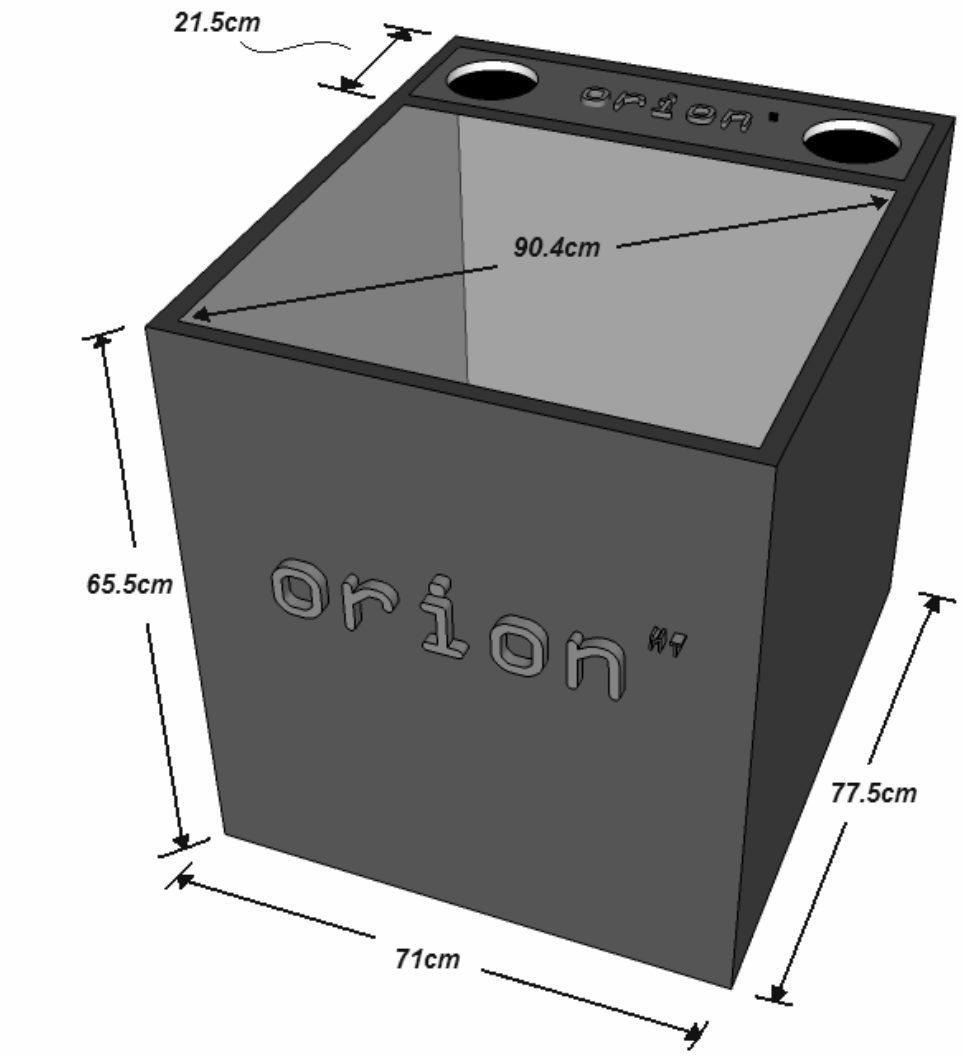


图 2：封闭箱子示例

这个箱子（图 2 所示）用 12mm 厚的中密度木板制作。在箱子的上部有一个格架，可以放镜子，增加投影的投射距离。这个格架还可以用来安置小喇叭或者用来放键盘。图 3 中的玻璃后来被换成一块 71x56cm，4mm 厚的单面打磨毛玻璃，作为投影面和散射体来用。

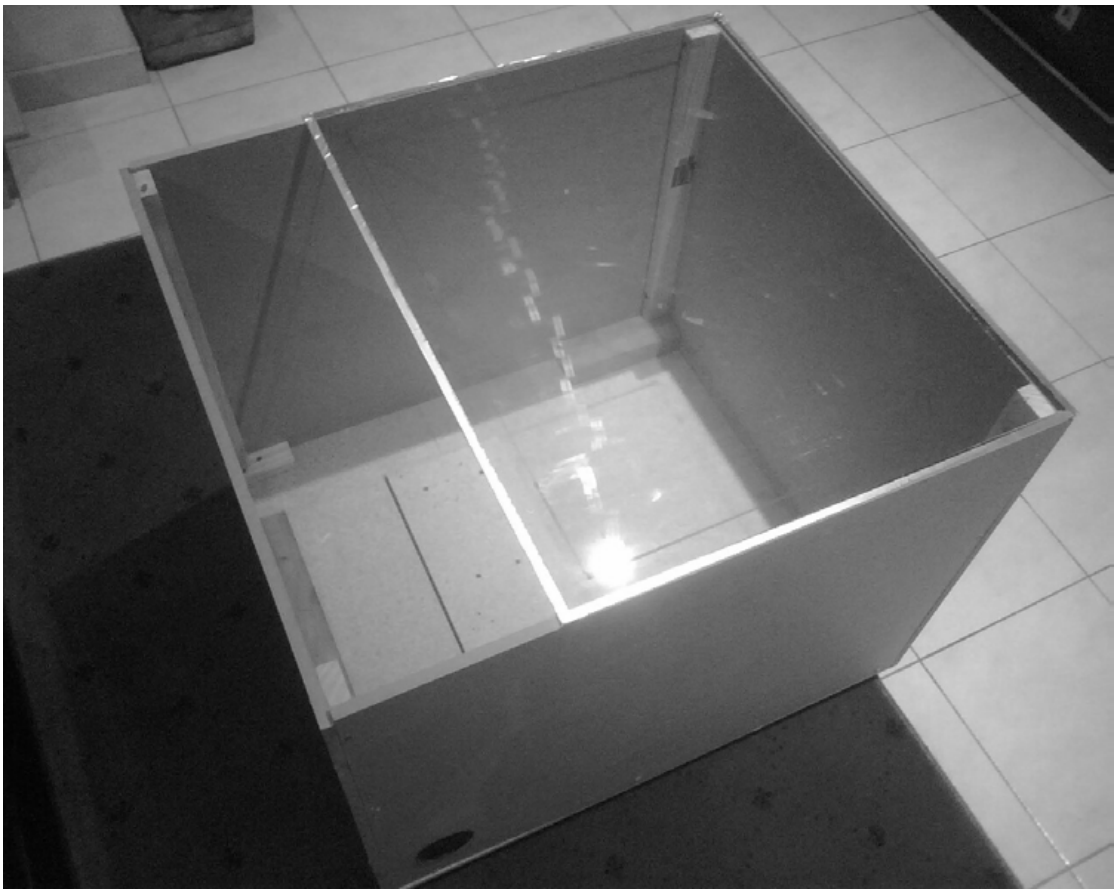


图 3：基于 3D 模型的半成品

第二步：投影仪

SHARP DLP 投影，型号：PG-M15S。

第三步：摄像头

Xbox360 USB 摄像头。分辨率：320x240@60fps。这个摄像头可以很方便的拆卸红外截止滤镜，价格也便宜，很适合做多点触摸摄像头。

第四步：红外照明

在 ORION 多点触摸桌中采用了 3 个光源(1x140 红外 LED, 1x80 红外 LED, 1x45 红外 LED)。具体分部如图 4-1，图 4-2 所示。图中圆形区域标出了各种照亮的区域。

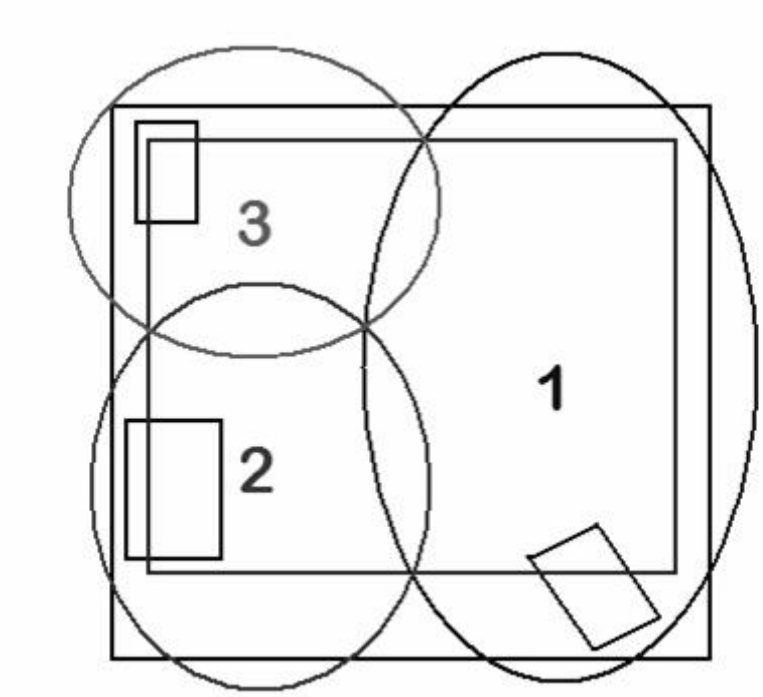
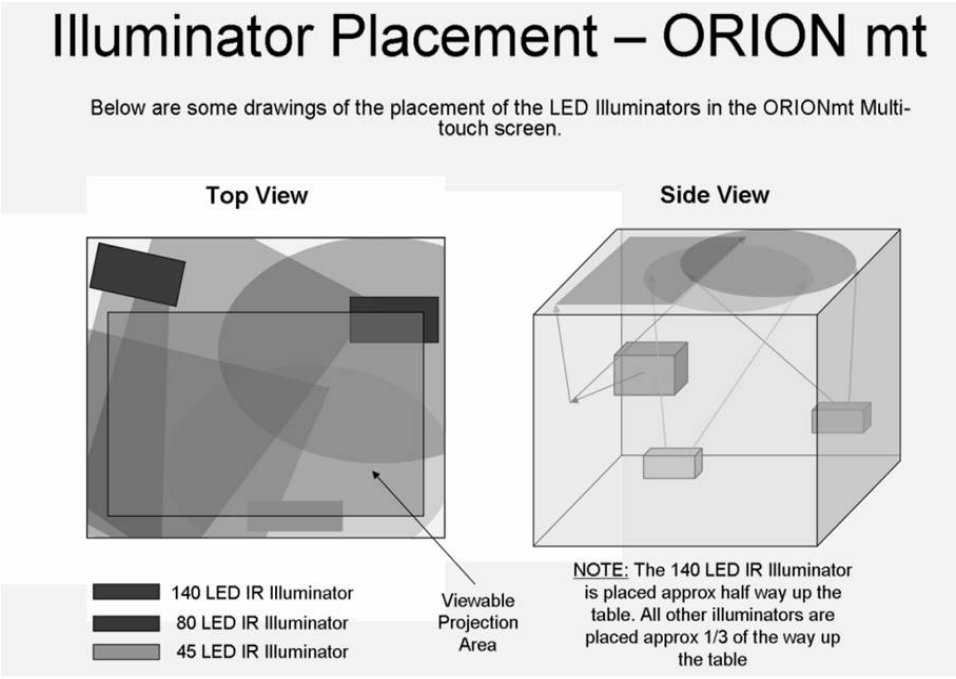


图 4-1：光源的光照区域

由于红外光源的发射角比较小，会在屏幕上形成一个高亮区域。所以 140 个红外 LED 光源转了一个角度，通过反射来照亮屏幕。

具体的放置方式参考图 4-2：



在图 4-2 所示的设置中，各个光源的照明区域是有重叠的，以保证整个触摸屏都布满红外光线。

图 5 是一些 140 个红外 LED 灯的细节：

图 5：DI 设置—投影仪，光影，镜子

·内置的感光器（被盖住了）

·照明范围：距离 80m，角度 60 度（户外）

·额定功率：18W

·波长：850nm

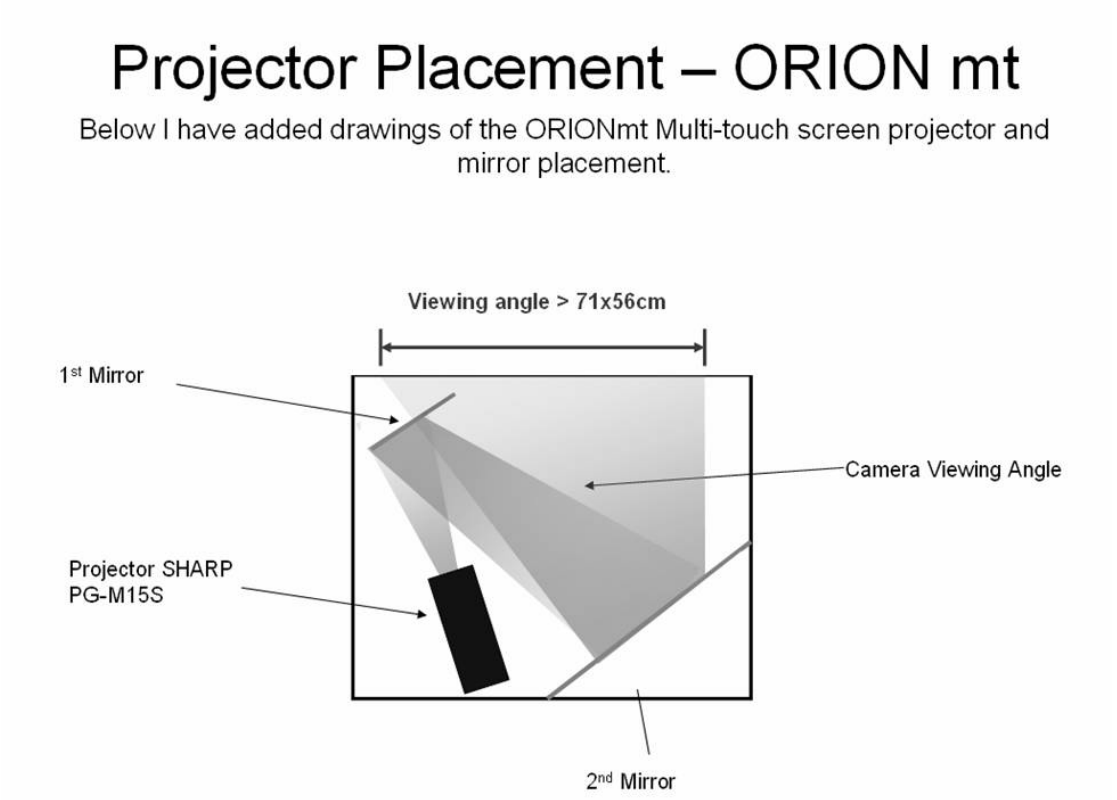
·电源：12V 直流 1000mA

80LED 和 45LED 光源是我们根据我们设计的 LED 计算器自己做的。如有需要，请访问 http://led.linear1.org/led.wiz

一开始，只用了 140 个 LED 的光源。但是光线太弱，以至不能形成清晰的

点，这意味着需要更多的红外照明，所以，又加了两个光影（2 和 3）。

第五步：摄像头和投影仪的位置



摄像头，镜子和投影仪的位置非常重要，要保证摄像头可以捕捉整个屏幕区域，投影仪可以将画面投到整个屏幕。一开始，摄像头被放在底部的中间，发现没法捕捉整个区域，所以通过镜子进行一次反射，这样整个屏幕就可以被捕捉到了。

第六步：触摸表面

你需要一个足够厚的触摸屏，而且受按压不会变形，另外还需要一个投影幕和散射体。一般采用的有描图纸（硫酸纸），桌布甚至浴帘（在我的方案中，这个效果最好）放在触摸屏上面，在购买投影介质前最好测试一下耐用性，投影成像质量和红外线散射效果。

图 6：Touchlib（触点捕捉软件）截图—调节模式

第七步：最终产品

下一步是用触点捕捉软件进行测试。如果你仔细观看图 6，你会发现 3 个被照亮区域，这正是 3 个光源所形成的区域被摄像头捕捉到后的原始画面（前两个窗口）

附录 D：制作一个可交互的地面

这一章将介绍如何建一个可交互的地面。这个交互地面的最大特点就是可以同时与多个用户互动，而不像其他交互地面，一次只能与一个用户互动。典型的情况是让地面的一部分可以进行互动，比如 lift lobby，或者在入口处，因为要让整个地面都能互动虽然可以实现，但没有太大的实际意义。

要注意一点的是，不要把这个作为你的第一个多点触摸项目。因为在做这样的项目前，你需要一些洞察和处理给种问题的经验。

根据房间的不同情况和安装人员的水平，这个项目将需要 4-8 小时。资金投入取决于房间的大小和相应的元件。这没有一个确切的数字。

在正式开工之前，实地考察一下，看能否把设备都隐藏起来，只留一个小孔 。还有，房间里的照明情况是首要考虑的因素。如果太靠近窗口而且有阳光干扰的话，很可能会影响到效果。如果可能，可以在窗户上蒙上一层材料，阻挡红外线 。

你也许注意到没有红外光源。原因是 95%的 lobby 区域都有照明装置，这些照明装置都可以均匀的发射红外线。如果 lobby 没有光源的话，你可以很容易的在摄像头旁边加装光源。

地板的亮度一般不成问题。实际上，在某些情况下还能帮助进行追踪。简而言之——地板越暗，效果越好。如果地板不是那么亮，或者反光，别担心，那就换成白色地板，白色地板最适合投影了。

警告：由于所有的设备都固定在天花板上，要确保这些设备不会掉下来，造成不必要的伤害。

不同的技术方案:

基本上，有两种方式。第一种是把设备装到天花板里面，第二种是把设备装在天花板下面。我们先讨论第一种方案，然后再讨论第二种。如图 1.

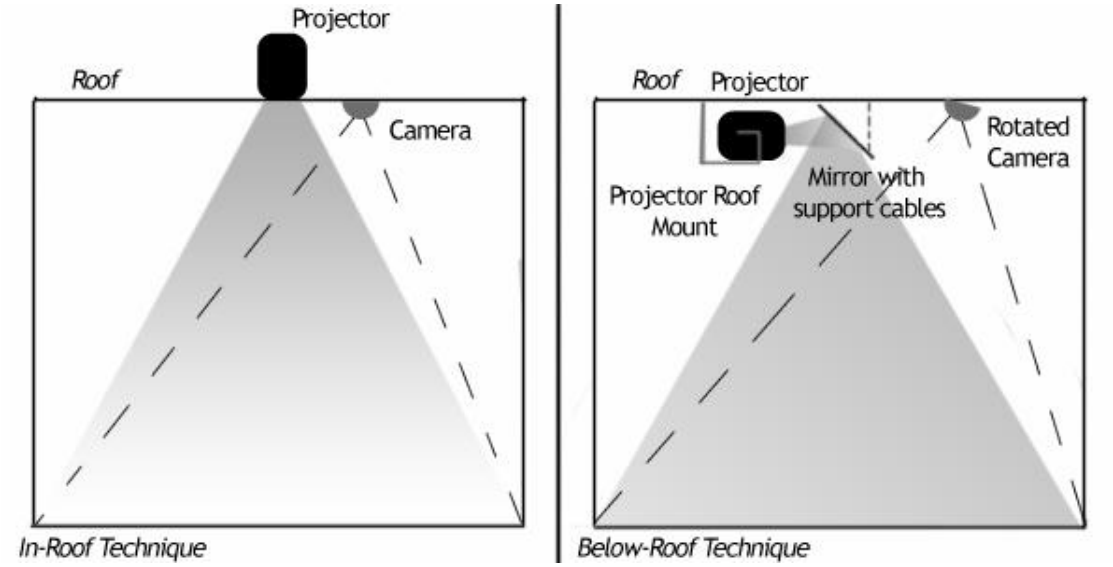


图 1：两种方案对比

不同的方案需要的工具也不一样。下面的列表仅供参考。

·投影仪：让投影尽可能的亮一些。同时采用广角镜头或者短焦投影，以投出更大的画面。

·摄像头：用带有广角镜头的摄像头，以确保捕捉整个投影画面。

·锤子和螺丝刀：根据具体情况而定。

方案一

步骤:

警告: 用一个比较稳定的梯子。找一个人辅助你，要不投影落下来会伤着你 。

第一步：固定投影仪

首先，你需要在天花板上打一个洞，洞的大小要适合投影仪的镜头。这要根据具体情况而定。这一步的关键是投影仪要放在这块天花板的正中间。同时还要

保证投影区域要和你的要求吻合。

第二步：安装摄像头

推荐使用 CCTV(闭路电视)之类的摄像头，因为他们适合安装在天花板上。如果有支架，用螺丝钉把他们固定在天花板上。如果没有支架，则需要额外买一个。

第三步：安全检查

在继续下一步前做一个测试。找一个人帮助你完成者一步。

摇一下摄像头，看是否安全。如果不安全，就再加固，否则，继续。

摇一下投影仪，看是否安全。如果不安全，就再加固，否则，继续。

通常情况下，安装好之后，过一夜，第二天再检查一次，确保安全。

第四步：测试

确保所有的安装完成，准备测试。具体的接线方式你自己决定，一般来说，需要延长线。

安装软件，测试，根据需要进行调整。

方案二：

工具列表，仅供参考。

·投影仪：尽可能的轻。同时采用广角镜头或者短焦投影，以投出更大的画面。

·摄像头：用带有广角镜头的摄像头，以确保捕捉整个投影画面。

·锤子和螺丝刀：根据具体情况而定。

·投影支架：尽可能轻，但坚固。

·镜子：任何镜子。

·镜子支架：一般来说金属支架可以适合任何表面。

步骤:

警告: 用一个比较稳定的梯子。找一个人辅助你, 要不投影落下来会伤着你 。

第一步：

根据要求，固定投影仪，镜子支架，镜子。

第二步：安装摄像头

推荐使用 CCTV 之类的摄像头，因为他们适合安装在天花板上。如果有支架，用螺丝钉把他们固定在天花板上。如果没有支架，则需要额外买一个。

第三步：安全检查

在继续下一步钱，做一下测试。找一个人辅助你。

摇一下摄像头，镜子和投影仪，看他们是否安全。如果不安全，加固他们，否则继续。

第四步：测试

确保所有的安装完成，准备测试。具体的接线方式你自己决定，一般来说，需要延长线。

所用软件

为了校准系统，你可以采用 touchlib 或者 Community Core Vision (CCV).你需要设置你的软件，只侦测大型的点。这样，就能选择性的捕捉到地板上的人。

测试软件的时候, 要尝试在不同的光线条件下进行。不同的光线会影响到具体的设置。推荐的时间是中午 12 点左右。这对成天使用的产品来说会得到最好的精度。

下面的步骤是用 CCV 或者 Touchlib 进行校准。

启动 CCV 或 Touchlib

拿一个手电筒

点亮手电筒，照在地面上，调节 CCV 或 Touchlib 的滤镜，让它只看到手电筒的光。换句话说，把 rectify 值设得比较高。

关掉手电，在 CCV 中重新捕捉背景（按 B 键）。

按 C 键进入校准界面，再按 C 键，开始校准。

把手电筒对准带绿色十字架的红色圆环。

快速的亮灭手电，越快越好。CCV 或 Touchlib 会把手电筒的闪光当作触摸操作。

重复第 6 和第 7 步，直至校准完成。

校准完成。

校准时的注意事项:

站在摄像头的捕捉范围之外，否则会干扰校准。

让手电筒的光亮尽可能的与校准点的位置重合。如果手电的照射角度太小，光斑会成椭圆形，这也会影响校准的精度。

总结

这个设置基本上还算简单。实际上每个设置都不同，你需要多实践，找到最适合的方式和步骤。这里所谈到的概念基本上都可行。

在做这种项目之前,最好有一定的多点触摸经验,这会帮你很快的找出问题，节省时间，也节省钱。

附录 E：参考资料

这些参考资料安装章节来组织

Actionscript 3 (Flash)

[1] Adobe Flash, Wikipedia entry. http://en.wikipedia.org/wiki/Adobe_Flash

[2] “Flash for surface computing” by Manvesh Vyas
http://www.adobe.com/devnet/edu/articles/manvesh-vyas.html

[3] flosc: Flash Open Sound Control http://www.benchun.net/flosc/

[4] Migrating from ActionScript 2.0 to ActionScript 3.0: Key concepts and changes by Dan Carr
http://www.adobe.com/devnet/flash/articles/first_as3_application.html

[5] AS3: Dictionary Object by Grant Skinner
http://www.gskinner.com/blog/archives/2006/07/as3_dictionary.html

Python

[1] Python Programming Language. http://www.python.org

[2] PyMT. A Python module for writing multi-touch applications.
http://pymt.txzone.net

[3] Zelle, J. M., “Python as a First Language,” Proceedings of the 13th Annual Midwest Computer Conference, March 1999. Also available at:
http://mcsp.wartburg.edu/zelle/python/python-first.html

[4] touchPy. http://code.google.com/p/touchpy/

[5] PointIR YouTube demonstration. PyCon 2008. http://www.youtube.com/watch?v=bEf3nGjOgpU

[6] libAVG. A high-level media development platform. http://www.libavg.de/

[7] pyTUIO. A python module for the TUIO protocol. http://code.google.com/p/pytuio/

[8] Kaltenbrunner, M., Bovermann, T., Bencina, R., Costanza, E.: "TUIO - A Protocol for Table Based Tangible User Interfaces". Proceedings of the 6th International Workshop on Gesture in Human-Computer Interaction and Simulation (GW 2005), Vannes, France, 2005. see also: http://www.tuio.org/

[9] Teiche, Alex. http://xelapondsstuff.wordpress.com/

[10] Hansen, Thomas. http://cs.uiowa.edu/~tehansen

[11] Pyglet. http://www.pyglet.org

[12] OpenGL http://www.opengl.org/

[13] Wikipedia: OpenGL http://en.wikipedia.org/wiki/OpenGL

[14] OpenGL Architecture Review Board, Dave Shreiner, Mason Woo, Jackie Neider, Tom Davis. OpenGL Programming Guide: The Official Guide to Learning OpenGL. ("The Red Book"). (Amazon link: http://www.amazon.com/exec/obidos/ASIN/0321481003/)

[15] http://nehe.gamedev.net/

[16] Wikipedia. GUI Widget. http://en.wikipedia.org/wiki/GUI_widget

[17] PyMT API Documentation. http://pymt.txzone.net/docs/api/

[18] Wikipedia. Transformation Matrix .

http://en.wikipedia.org/wiki/Transformation_matrix

[19] Affine transformations. http://www.leptonica.com/affine.html

[20] Wikipedia. Angle http://en.wikipedia.org/wiki/Angle

[21] Wikipedia. Euclidean Vector.

http://en.wikipedia.org/wiki/Euclidean_vector

Gesture Recognition

[1] Grafiti : Alessandro De Nardi

[2] Real-time gesture recognition by means of hybrid recognizers : Corradini Andrea

[3]The Biological Foundations of Gestures: J. Nespoulous, P. Perron, A. R. Lecours.

[4] A Hidden Markov Model-Based Isolated and Meaningful Hand Gesture Recognition : Mahmoud Elmezain, Ayoub Al-Hamadi, J¨org Appenrodt, and Bernd Michaelis

Miscellaneous

[1] Han, Jerfferson Y. “Low Cost Multi-Touch Sensing through Frustrated Total Internal Reflection.” Symposium on User Interface Software and Technology: Proceedings of the 18th annual ACM symposium on User interface software and technology. Seattle,WA, USA, 2005. 115-118.

[2] Gettys, Edward W, Frederick J Keller, and Malcolm J Skove. Classical and Modern Physics. New York: McGraw-Hill, 1989.

[3] Real-time gesture recognition by means of hybrid recognizers : Corradini

Andrea

[4] The Biological Foundations of Gestures: J. Nespoulous, P. Perron, A. R. Lecours.

[5] A Hidden Markov Model-Based Isolated and Meaningful Hand Gesture Recognition: Mahmoud Elmezain, Ayoub Al-Hamadi, J¨org Appenrodt, and Bernd Michaelis

这本《多点触摸技术》凝聚了 18 位专家的智慧，50 多幅图片，页数达到 100 页[英文版，译者注]，提供了丰富的实战经验和理论知识，给多点触摸领域带来全新的观念。作为这个领域的第一本出版物,作者们努力使这本书囊括最新的资讯,洞察和研究成果,并且经得起实践的检验.

这本书涉及但不限于以下领域:

·建造多点触摸系统的不同方法和工具

·多点触摸系统的原理

·在交互，设计和用户体验方面的挑战

·开源程序，组织框架（frameworks）和开发类库（libraries）

·在这个领域的公司和组织

中文版历经将近两个月，在翻译过程中虽说遇到高温和其它预想不到的情况，但是最终经过翻译组的努力，为大家奉上这本国内第一本也是唯一一本介绍多点触摸及其相关技术的书籍《多点触摸技术手册中文版》

中文版将分为简体中文版和繁体中文版，以符合不同地区的交流习惯。

JOIN US!