软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2019,30(Suppl.(2)):48−61

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于混合输入的场景设计工具*

仝青山[1,2], 张宗琦[1,2], 黄 进[1], 田 丰[1], 刘 杰[1], 戴国忠[1]

[1](人机交互北京市重点实验室(中国科学院 软件研究所),北京 100190)

[2](中国科学院大学 计算机科学与技术学院,北京 100049)

通讯作者: 田丰, E-mail: tianfeng@iscas.ac.cn

摘 要: 笔式用户界面作为 Post-WIMP 界面中的一种,以触控技术为依托,摒弃了物理键盘和鼠标,在一定程度上改变了人机交互的方式.草图绘制和识别软件不断涌现,但是却一直没有成熟的笔式界面设计开发工具.基于 PGIS 交互范式,利用场景设计方法,开发了基于笔式交互原语的图形和草图混合输入的场景设计工具 SDT.首先,基于软件工程领域的高内聚低耦合原则提出了"分离-融合"设计方法,并据此提出了系统的总体架构;其次,从界面形式化描述、笔式交互原语和单字符、混合输入这 3 个方面介绍了关键技术;再次,通过一个完整示例对该工具进行了更具体的展示,同时佐证了该系统的可用性和可行性;最后,通过两个评估实验,验证了该工具的先进性和有效性.

关键词: 笔式交互;场景设计;混合输入;交互原语;分离融合

中文引用格式: 仝青山,张宗琦,黄进,田丰,刘杰,戴国忠.基于混合输入的场景设计工具.软件学报,2019,30(Suppl.(2)):48−61. http://www.jos.org.cn/1000-9825/19017.htm

英文引用格式: Tong QS, Zhang ZQ, Huang J, Tian F, Liu J, Dai GZ. Scenario design tool based on hybrid input. Ruan Jian Xue Bao/Journal of Software, 2019,30(Suppl.(2)):48−61 (in Chinese). http://www.jos.org.cn/1000-9825/19017.htm

Scenario Design Tool Based on Hybrid Input

TONG Qing-Shan[1,2], ZHANG Zong-Qi[1], HUANG Jin[1], TIAN Feng[1], LIU Jie[1], DAI Guo-Zhong[1]

[1](Beijing Key Laboratory of Human-Computer Interaction (Institute of Software, Chinese Academy of Sciences), Beijing 100190, China)

[2](School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100190, China)

Abstract: Pen-based user interface relying on touch technology is one of the Post-WIMP interfaces. It discards the physical keyboard and mouse, which changed the method of human-computer interaction to some extent. Though sketch drawing software and recognition software are constantly emerging, there is no mature pen-based interface design development tool. Based on the PGIS interaction paradigm and scenario design method, this paper develops a tool named with SDT that allows hybrid input of graphics and sketchs based on pen-based interaction primitives. Firstly, based on the principle of high-cohesion and low-coupling in the field of software engineering, the "Separation-Fusion" design method is proposed. Accordingly, the overall architecture of the system is put forward. Secondly, the essential technologies are elaborated from three aspects: user interface description language, pen-based interactive primitive and mono-case, hybrid input. Thirdly, an example of a complete application is built by the SDT, which makes the availability and feasibility of the system more convincible. Finally, the advantage and effectiveness of the tool are verified by two evaluation experiments.

Key words: pen-based interaction; scenario design; hybrid input; interactive primitive; separation-fusion

笔式用户界面(pen-based user interface,简称 PUI)是 Post-WIMP 界面的一个主要形态,它通过纸笔(paper-pen)隐喻定义了人和计算机之间一种更符合人类习惯的自然交互方式[1].人类大脑的创造性思维和灵感,基本

* 基金项目: 国家重点研发计划(2016YFB1001405); 国家自然科学基金(61802379)

Foundation item: National Key Research and Development Program of China (2016YFB1001405); National Natual Science Foundation of China (61802379)

收稿时间: 2019-07-15; 采用时间: 2019-11-04

上都是始于纸笔;类似于原画设计、界面设计和需求调研等很多工作也都是利用纸笔来完成的.基于纸笔隐喻的笔式交互,人的注意力都集中在思维和创作上,可以很自然地利用纸笔进行自然地勾勒;传统的 WIMP 界面则会把人的注意力分散在屏幕上不断闪烁的光标,而为了记录信息手则不断穿梭于鼠标与键盘之间[2].笔式用户界面作为一种自然、高效的交互方式,在信息捕捉、信息交流、艺术创作和设计制造等多个领域所应用,并且随着触屏设备的普及,它的需求也越来越大.

基于笔式用户界面软件在不同行业、不同领域的大量需求,单独为每一个客户开发应用给公司造成了巨大压力,也因一些重复工作造成了成本的额外花销,这就对通用笔式用户界面软件设计工具提出了迫切需求.有不少学者和公司开始致力于笔式设计工具的开发,也产生了不少产品,但这些产品大都是原型系统,而且基本都是草图设计工具.另外,很多软件只是增加了对笔的支持,用笔、手指的触摸代替了鼠标键盘,并未对笔所特有的交互特性进行扩展.目前,设计师使用的设计工具基本都是界面设计工具,而且多是界面的 web 呈现,不能直接生成用户可用的终端应用.中国科学院软件研究所人机交互实验室在笔式交互平台上做了大量研究,提出了 PGIS 范式,也开发了诸多应用.

本文就是基于笔式界面设计工具的现状,以 PGIS 交互范式为指南,利用已有的交互引擎,结合场景设计方法,提出了“分离-融合”的设计方法,构建了系统的总体架构,并实现了该系统.通过一个完整实例更完整地展现了该工具的独特性和可用性,用过两个对比实验对用户表现进行了评估,验证了该工具的有效性和先进性.该工具不仅能够构建用户界面原型,而且生成后的终端应用可以动态添加界面元素和其他内容信息.该场景设计工具具有以下特点.

(1) 同时支持传统笔式交互原语和草图两种输入方式设计用户界面;
(2) 以自顶向下的思想构建应用;
(3) 以状态迁移图的形式展示界面间的关系,并进行交互动作和交互行为的设置;
(4) 面向最终用户,无需编码直接生成应用程序;
(5) 交互对象和应用对象彼此独立,在生成的应用中才建立联系;
(6) 生成的应用程序可以具有设计工具本身所不具有的功能.

1 相关工作

用户界面设计工具的研究一直是计算机领域的研究重点之一.使用设计良好、简单易用、贴近用户自然使用习惯的用户界面设计工具会极大地提高用户的工作效率.在工业界有很多优秀的用户界面设计工具,例如 Axure、POP 等.这些工具具有丰富的用户界面元素库,并提供了多种多样的功能用于用户界面的设计.

用户界面设计工具按照交互所使用的工具来说分为两种:一种是使用鼠标的 tool-based 设计工具,另一种是使用触控笔的 sketch-based 设计工具.有些研究者关于这两者之间的差异做了一些研究,参考文献[3]中指出,使用笔交互具有更高的效率,但是使用鼠标交互拥有更好的便利性.在 sketch-based 的设计工具中,使用识别算法识别用户在屏幕上绘制的草图是普遍使用的方法.

在用户界面的设计过程中,草图是一种十分有效的表达设计意图的途径[4].国内外有许多关于草图的用户界面设计工具的研究.CMU 提出的 SILK[5]是一款基于草图的界面设计工具,该工具支持用户使用触控笔或鼠标在屏幕上进行手绘的操作,通过草图识别算法,快速地生成可交互的用户界面元素.SILK 所做的贡献可谓是开创性的,但是 SILK 在设计之初并没有充分地考虑到笔式交互的连续性特点,而仅仅是将笔作为鼠标的替代工具,其设计思想仍是基于 WIMP 交互范式的,在后期的精细设计中,仍然需要用户通过鼠标点击等传统方式进行设计.

Penbuilder[6]是一个支持笔式交互的用户界面设计工具.相对于其他笔式交互用户界面设计工具[7],Penbuilder 是完全基于 PUI 的用户界面设计工具.Penbuilder 将纸笔隐喻作为其设计思想的基础,在整个设计的各个方面都对笔式用户界面进行了针对性的设计,使得 Penbuilder 拥有强大的笔交互能力.Penbuilder 针对笔交互的特性建立了特别的事件模型,使用事件分析树的形式区分用户的绘制与手势操作,使得用户在使用的过程

中无需切换工具即可快速的表达设计思想,从而构建用户界面原型.

在人机交互领域,场景这一概念描述的是人与人活动关系的故事[8].基于场景进行软件的界面开发是一种效率较高的开发方式[9].PUI maker[10]是基于场景的思想指导开发出来的笔式用户界面工具.PUI Maker 使用笔交互进行输入,基于 PUIML 进行用户界面的描述,使用户界面开发人员可以通过纸笔交互的方式方便而自然地合作实现用户界面的开发工作.PUI Maker 解决了 SILK 只能使用笔交互完成用户界面的初期设计,而不能完成用户界面整体开发的问题.PUI Maker 使用场景树描述应用程序不同界面的跳转关系,使用状态迁移图描述用户在某个特定界面上的交互动作.对于在用户界面上如何定义用户的交互动作的相关研究也有很多,UISKEI[11]就是一个基于笔交互的用户界面设计与评估工具.UISKEI 定义了功能十分强大的交互设计工作,但是此功能必须使用键盘与鼠标进行操作,十分不利于用户在设计用户界面时快速的探索思路.

2 场景设计工具

本文的场景设计工具(scenario design tool,简称 SDT)是基于场景的软件设计方法,利用自顶向下的开发思想,构建了由场景树到场景的设计流程;同时,利用系统各部分弱耦合的思想,提出了“分离-融合”的设计方法,完成了集界面设计、动作设计、代码生成和自动编译为一体的软件设计工具.

2.1 基于场景的软件设计方法

场景的概念在不同领域有这不同的含义,本文中的场景是指一个故事片段,对应话剧中的一幕,是关于人及人的活动的故事,操作者(演员)、操作环境(舞台)、操作目的(意图)、行为活动(动作)是场景的4个要素.基于场景设计(scenario-based design)的思想隐式对系统的操作进行了限制.它将设计工作的焦点定位在描述什么人使用该系统去完成目标任务,是提高软件可用性的一种有效途径[12],同时也是提高用户界面开发效率的一种方法.

在软件设计人员眼中,一个场景就是一个用户界面,场景里的饰物(道具)就是界面元素(控件),场景中的人就是软件的操作者,场景中人的活动就是软件的动态交互行为;整个场景树就是多个用户界面的结合体,按照树图结构进行跳转和切换.如图 1 所示,S0 是应用程序启动页面,用户可以通过与页面 S0 上的元素交互行为跳转页面 S1 或 S2,但是用户并不能直接从 S0 跳转到 S11、S12、S21 和 S22,只能按照节点之间的连接进行跳转.

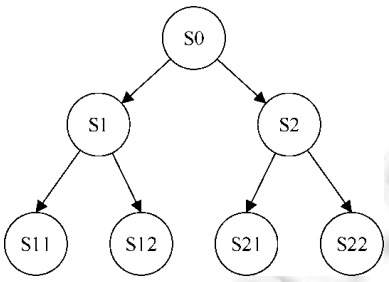


Fig.1 Scene tree structure diagram

图 1 场景树结构图

2.2 “分离-融合”设计方法

耦合度和内聚度是软件设计中衡量模块独立程度的重要度量特征,高内聚低耦合,已经成为软件工程领域判断设计好坏的标准.“分离-融合”设计方法是在此基础上发展而来,该思想的示意图如图 2 所示.场景设计工具 SDT 直接调用 PGIS 交互引擎的 API,来开发各个编辑器和其他功能模块.在 SDT 整个运行周期,都没有调用 CDM 函数,它们之间是分离的关系;但是在场景动作设计阶段通过 XML 任务列表又与 CDM API 的字符串形式存在着微弱的关联.SDT 正是利用这种微弱的关联,待用户完成了全部设计后在生成模板的作用下生成了目标代码,经过编译器的编译生成了最终的应用程序 App.生成的 App 直接调用了 PGIS 和 CDM 的 API,实现了 PGIS

交互对象和CDM应用对象的融合.这种设计阶段分离、App生成后融合的思想被我们称为“分离-融合”设计方法.利用这种设计方法,当CDM增加新的组件或功能后,SDT仅需更换任务列表文档和CDM类库,无需更改代码就可以直接使用,减少了模块间的依赖.

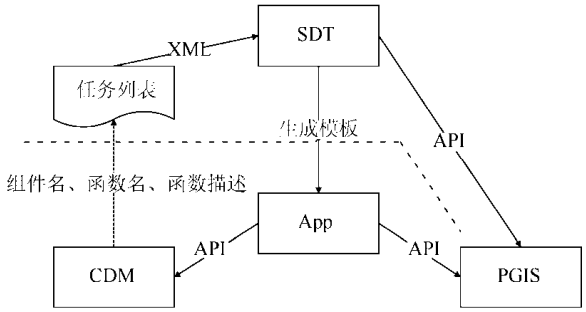


Fig.2 “Separation-Fusion” design diagram

图2 “分离-融合”设计示意图

2.3 系统总体架构

本文的场景设计工具SDT是面向最终用户的,主要由4个模块组成:界面形式化描述与数据存储、界面编辑模块、代码生成模块和代码编译模块,如图3所示.

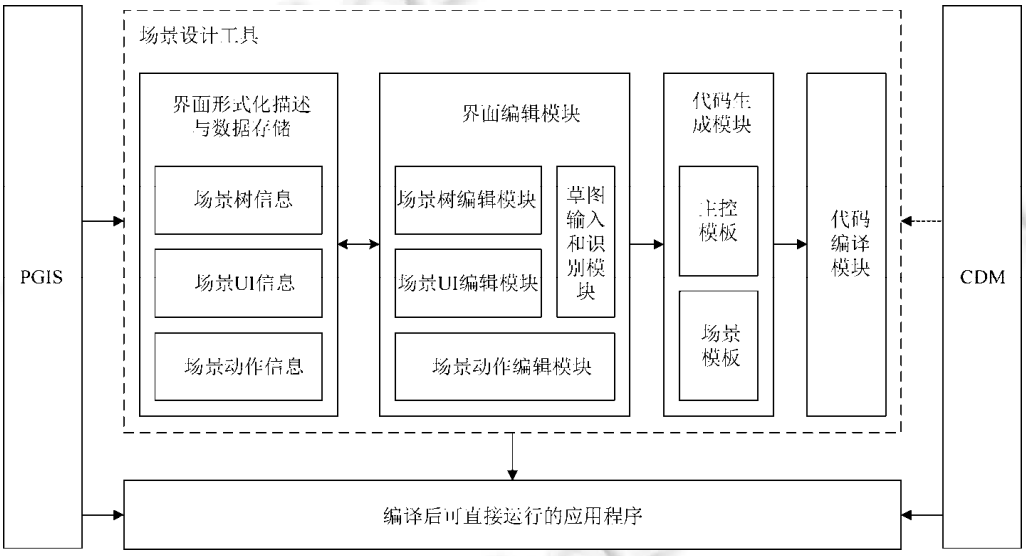


Fig.3 System overall architecture

图3 系统总体架构

PGIS交互引擎模块是基于PGIS(paper,gadget,icon,sketch)交互范式构建的软件平台.它封装了底层的显示引擎、媒体引擎、Ink引擎,为上层应用提供了交互对象的描述、交互对象的管理、交互对象的显示和交互对象的消息响应.交互引擎模块定义了诸如Icon、Static等交互对象的数据结构,包括其位置、大小、形状、背景图片、背景颜色、文本内容、文本大小、前景颜色、字体、显示状态等.此外,交互对象还对常见的笔手势进行了描述和管理,如删除等操作.交互引擎对各种交互对象提供了统一的接口,对交互对象的创建、释放等操作进行统一管理.交互对象的显示是用户界面的根本需求,交互引擎根据用户的输入参数,提供多种状态的显示效果.交互对象的消息响应是交互引擎的核心.它对系统层的事件进行了封装,对外提供了短击、长击、拉框、拖

框和草图这 5 种交互原语,用于用户和界面交互对象的交互.

CDM(card document manager)是基于笔的卡片文档数据管理系统.它以卡片(书)为原型,用页(Page)、层(Layer)、框(Frame)这种三级类 DOM 树的形式组织、存储数据和显示数据.页、层、框是利用底层 Ink 引擎、媒体引擎、显示引擎和交互引擎构建的应用对象,每个应用对象按照功能进行封装,对外以固定的格式提供相应的 API,以供 SDT 生成的应用程序调用.

场景设计工具直接利用 PGIS 交互引擎提供的 API,构建并显示各种交互对象;CDM 不是直接提供 API 供 SDT 调用,而是以 xml 文件形式提供应用对象的功能函数(场景任务)列表供 SDT 使用.用户按照自己的意图设计完场景树和各场景界面 UI 后,对每个场景的某些交互对象设置其动态交互行为,主要包括用户对该对象进行交互原语种类的选择和进行交互动作(主要包括场景迁移和场景任务两种)的选择.通过场景迁移可以完成从一个场景界面到另一个场景界面跳转的设置,通过场景任务可以为交互对象绑定一个 CDM 应用对象提供的功能,待生成代码并编译生成应用程序后即可执行相应的动作.

界面形式化描述是与用户界面上每一个交互对象的显示状态一一对应的,通过界面的形式化描述使得交互对象的数据结构与 XML 数据存储始终保持一致;同时,本文的界面形式化描述还囊括了场景动作信息,使得数据的存储更加完备.

界面编辑模块主要包括 3 部分:场景树编辑、场景 UI 编辑和场景动作编辑.其中,场景树编辑模块是用户对系统整体功能的一个概要设计,而设计思想则体现在用户添加了多少节点以及节点之间的连接上.场景 UI 编辑模块是用户对该界面布局的一个详细设计和实现,反映了用户的真正需求.场景动作编辑模块则是利用场景树构建的场景迁移图,把用户界面之间的跳转关系通过交互对象进行了绑定,把交互对象在界面内可执行的交互原语和场景任务通过函数的形式进行了功能绑定.由于场景树和场景 UI 是用户思维的体现,为了让用户把主要精力集中在构思上,本文还提供了场景树编辑和 UI 编辑的草图版本;在草图输入界面,用户用笔输入指定的图形手势,笔提起时草图系统会利用 Dollar1 算法进行草图识别,识别完成后,系统会在草图区域绘制标准化的 UI 元素,具体如图 4 所示.

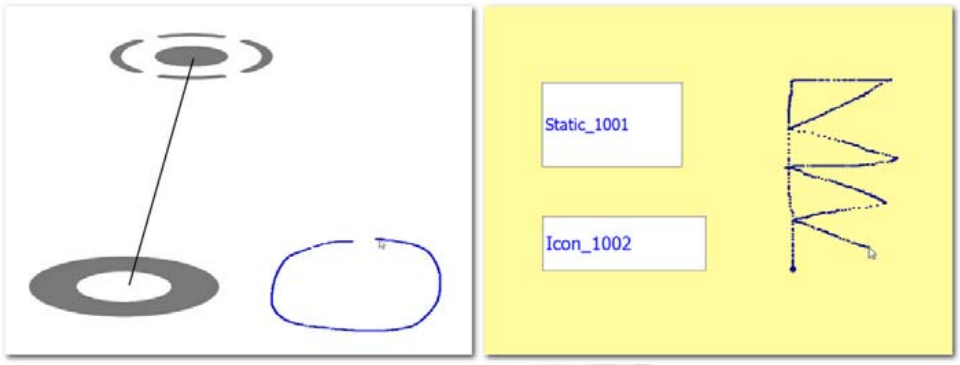


Fig.4 Sketch input of scene tree and scene UI

图 4 场景树与场景 UI 草图输入

代码生成模块主要把用户设计的场景树、场景 UI 和场景动作信息通过对预设模板进行数据填充和替换,最终生成应用程序代码.代码生成模块主要包括主控模板和场景模板.主控模板主要负责系统资源的管理,首先对交互引擎和 CDM 库初始化,为交互对象、应用对象以及笔的使用提供支持,然后进入根场景,加载场景树,最后按顺序释放系统资源.场景模板负责界面上各交互对象的初始化和其交互行为的设置,首先按照各类交互对象的模板得到其初始化代码,然后再把其填充到场景模板的相应位置,再检查该交互对象是否具有场景动作,再利用场景迁移或者场景任务模板块把相应代码嵌入场景模板中.

代码编译阶段的主要工作是编译脚本 makefile 的动态生成和脚本的执行.首先,根据源代码存在的路径生

成目标代码列表,然后设置外联的类库目录及类库名称,然后设置编译的最终目标、编译参数和连接参数,最后通过编译器的编译命令把上述各种参数连接在一起.待脚本完成后即可根据选择的编译器用 make、nmake 等命令进行最终的编译工作.

3 关键技术

3.1 界面形式化描述

数据是软件的灵魂,是软件运行的内在驱动力.通过用户与用户界面的交互,数据不断地产生和变化.软件设计工具存储的数据是用户界面信息和程序源代码.通过使用 UIDL(user interface description language,用户界面描述语言)来支持基于模型的用户界面的开发,是降低界面开发时间及复杂度的有效方法[2].用户界面描述语言在整个应用系统的开发中扮演了核心模型的角色.它通过一系列规范的语言对用户界面的各个组成部分进行描述.本文开发的设计工具用户界面主要包括 3 部分:场景树编辑界面、场景动作编辑界面和场景 UI 编辑界面,它们对应的 UIDL 描述如下.

场景树描述:

<场景树>::={<场景>}+{<连接>}

<场景>::=<场景代号>+<名>+<说明>

<连接>::=<连接代号>+<父代号>+<子代号>

场景动作描述:

<场景步>::=<场景代号>+{<交互原语>+<场景动作>}

<交互原语>::=<代号><交互类型><说明>

<交互类型>::=<单击>|<双击>|<拉框>|<拖框>|<划线>|<哑>

<场景动作>::=<代号><动作类型><说明>

<动作类型>::=<子场景代号>|<父场景代号>|<场景任务>

<场景任务>::=<代号><任务类型><说明>

场景 UI 总体描述:

<UI>::=<背景>{<交互对象>}

<背景>::=<颜色>|<图片>|<GadgetId>

<交互对象>::= <Icon>|<Slider>|<ITable>|<TTable>|<Piemenu>|<Palette>

交互对象描述(以 Icon 为例):

<Icon>::= <基本信息><图片><Icon 状态><Icon 边缘色><Icon 背景色><text 信息>…<矩形圆角率>

<基本信息>::=<类型><Id 号><包围框><是否响应><Gadget Id 号>

<图片>::= 图片在文件系统中的路径

<Icon 状态>::=常规|按下|不可用

<颜色>::= 颜色值(整形)

…

<矩形圆角率>::=浮点数

用户描述语言清晰地描述了对象的属性状态等信息,以它为原型进行数据的存储信息,同时创建程序运行时的数据结构,从而保证了硬盘存储的 xml 数据和内存中数据的一致性.

3.2 笔式交互原语和单字符

交互原语是用户通过交互设备与计算机之间的一个独立的、不可分割的最小操作.我们可用一个六元组来定义交互原语:*IP*=<*Task*,*Temporal*,*Spatial*,*Duration*,*Distance*,*Device*>.其中,IP 是 Interaction Primitives 的简称,

Task 表示基本的交互原语,Temporal 表示交互过程中的时间信息,Spatial 表示交互过程中的空间信息,Duration 表示从开始到结束的时长,Distance 表示从起始点到终点的距离,Device 表示产生原语的设备信息.

在笔式交互原语中,最基本的原语是笔迹(stroke),它代表用户在完成交互任务过程中输入的具有独立、最小和不可分割特性的一段笔式交互信息的集合.我们用 Stroke 来定义我们的交互原语,则可以表示为:*IP*=<*Stroke*,<*time*1,*time*2,…,*timeN*>,<*point*1,*point*2,...,*pointN*>,*Duration*,*Distance*,*Device*>. 交互原语的类型由画 Stroke 过程中的时间和空间特征决定,根据时间序列可以得到具体的时长,同样根据 Stroke 的空间坐标信息可以得到笔的运动轨迹和距离,然后分别与预定义的时长阈值 Duration、空间距离阈值 Distance 进行比较,就可以获知交互原语的类型.当 Duration 和 Distance 都非常小时,该交互原语可被视为单击 Tap;当 Duration 较长而 Distance 较小时,该交互原语可被视为长击 Hold;当 Duration 较小而 Distance 较大时,该交互原语可被视为拖框 AdjustFrame;当 Duration 和 Distance 都较大时,该交互原语可被视为拉框 DragFrame.此外,还可以从另一个角度来定义交互原语的类型,仍然可以得到同样结果.首先,根据时间长短将 Stroke 分为单击和长击两类,同样根据距离阈值将 Stroke 操作分为点、线两类;然后进行这两类之间的组合,就可以得到上述 4 种交互原语.除了标准的点、线之外,Stroke 还可以是曲线、折线等,它们被统称为 Sketch 交互原语.系统根据人们的行为习惯可以定义一些特定规则的 Stroke 为笔手势.

单字符是本文提出的一个概念.它的具体含义是用一个图符代表两种或多种含义,如图 5 所示.单字符的提出基于两个方面的原因:图符形式化表示方面和代码实现方面.第一,在进行场景动作设置时,需要将交互对象和交互原语进行绑定,但是从状态迁移图中很难同时看出是哪种类型的交互对象和交互原语.因此,为了使状态迁移图更能表达其含义,我们引进了单字符的概念,即用一个图标代表两种含义.第二,在程序实现时,对每个设置了场景动作的交互对象,我们会利用交互引擎获取的消息先行通过两层 switch/case(交互对象类型和交互原语)才能进一步定位该对象的场景动作,这样供需编码 $M\times N$ 次.实际上,并不是每个交互对象都拥有 5 种交互原语,如 ICON 就不支持 Sketch 操作,因此,我们对其进行了降维处理,把 $M\times N$ 种查找操作降到了 T 种操作,这一维的 T 种操作被称为单字符.另外,在代码层面,我们的单字符被定义成了枚举类型,如 *INTE_Icon_ACTIVE*=17,它是由交互对象类型 *INTE_Icon*=16 与交互原语 *Active*/*Tap*=1 综合的体现;示例还可以用二进制形式来表征:0*x*0001 0001=0*x*0001 0000|0*x*0000 0001,因此,我们不仅可以通过交互引擎的消息很容易地得到单字符,而且还可以通过单字符很容易地得到交互对象的类型和交互原语.

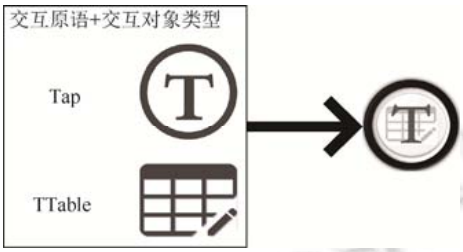


Fig.5 Monocase schematic diagram

图 5 单字符示意图

3.3 混合输入

本文的混合输入是指基于笔式交互原语的传统图形输入和智能化草图输入.草图输入源于笔式交互原语的基本概念笔迹(stroke),但又进行了智能化拓展.

笔迹是笔式交互原语中最基本的操作,我们根据其时间和空间信息定义的只占 Stroke 的一小部分,而其他的 Stroke 都被视为草图(sketch).在这些 Sketch 中,有一些操作与人们的行为习惯类似,意图很明显,我们把这种有目的持笔移动的行为称为手势.在本文的 PGIS 交互引擎中,把常见的 20 种操作封装成了手势,如从左到右的直线、从左到右的曲线、从右到左的箭头、从左上到右下的直线以及复杂的 z 字形折线.这 20 种基本手势与

交互原语不是孤立存在的,拖框和拉框的复杂交互原语就建立在其中的一些基本手势之上.因此,手势不是简单的笔迹数据,而且还能用于操作命令.

本文阐述的场景设计工具在进行界面设计时同时支持标准的笔式交互原语输入和特殊的草图手势输入.笔式交互原语方式可进行交互对象的创建和编辑、属性的设置和场景动作的绑定,还选用了一些手势用于连接线的绘制、对象的删除、返回父场景等操作.草图手势输入方式可以在场景树界面和场景界面进行草图输入与草图在线识别.用户可以基于传统的纸笔绘制操作,在编辑界面上用笔勾勒出对象的图形轮廓,提笔后草图识别引擎会利用 Dollar1[13]算法和已建立好的交互对象草图库识别出图形所代表的对象类型,然后系统会根据识别结果在原位置直接绘制目标对象.该工具在进行界面设计时的流程如图 6 所示.

Fig.6 Interface editing flow chart based on hybrid input

图 6 基于混合输入的界面编辑流程图

4 应用实例

本文基于场景的软件设计方法构建了满足 PGIS 交互范式的场景设计工具 SDT.该工具以可视化建模的方式提供了场景树、场景 UI 和场景动作的编辑功能,同时还提供了场景树和场景 UI 界面的草图输入与识别功能,最终结果都以 XML 格式进行存储.下面,我们将以儿童画板为例来讲述 SDT 工具的使用方法.

4.1 需求分析

儿童画板软件主要为儿童提供涂鸦、绘画和写字等功能.以纸笔为隐喻的笔式界面开发工具降低了儿童的认知负荷,可以实现由传统纸笔、实木或塑料画板到画板软件的平滑过度.

从功能分析角度来看,软件首要的功能是笔迹的实时绘制与编辑功能;其次,画笔的颜色要能够灵活切换;再者,画笔粗细应该可调节.此外,软件还应具有图形切分功能、纸张切换等功能.从视觉特征和用户心理特征角度来看,该软件应该颜色鲜艳、吸引眼球.从可用性角度来看,该软件应尽可能地简单、易用.

4.2 界面设计

实物儿童画板是可复用的画板,我们要设计的儿童画板软件把每一画版的内容视为画本中的一页,实物画板中人、物和文字与画本中的对象相对应.因此,我们利用自顶向下的思想,在利用场景树编辑器构建了“页—

框—具体框”的场景树,具体如图 7 左图所示,这样就可以在各个场景中通过交互对象建立与 CDM 中的应用对象建立连接.我们在页场景 UI 编辑界面的工作区中,新建了几个 ICON 对象,并对它的背景图片、文字信息等进行了设置,具体如图 7 右图所示.我们在页场景的场景动作编辑界面,通过状态迁移图对页面间执行跳转动作的交互对象和交互原语进行了设置,对界面内的若干交互对象通过选择哪种交互原语、执行哪种场景任务进行了绑定,具体如图 8 所示.

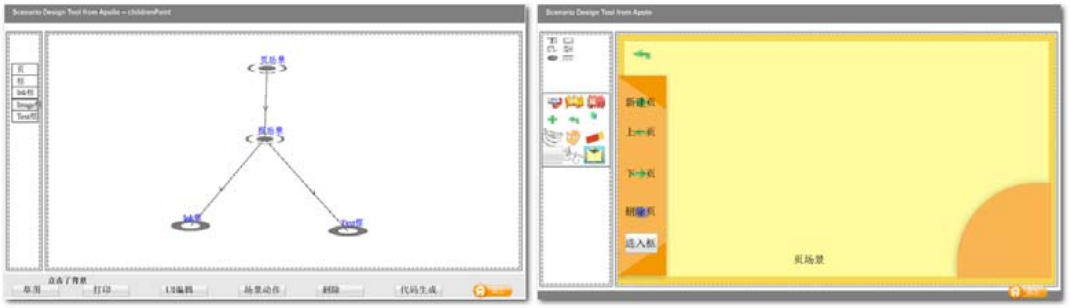


Fig.7 Page-frame scene tree editing interface

图 7 页框场景树编辑界面

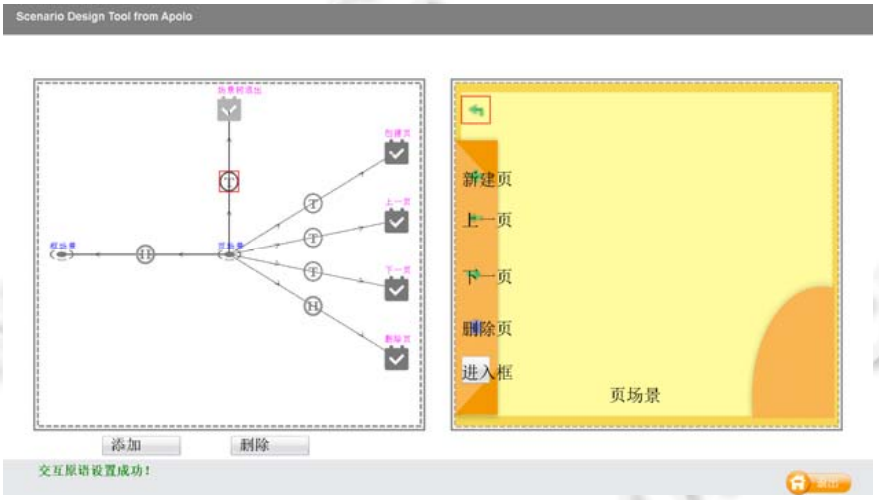


Fig.8 Page scene action editing interface

图 8 页场景动作编辑界面

4.3 生成应用

待所有的界面设计工作完成后,用户只点击代码生成按钮,系统就会依据内存或 xml 中存储的场景树、场景 UI 和场景动作信息把主控模板和场景模板具体化,生成相对应的源代码文件(.h/.c 文件);同时在生成代码的过程中生成编译脚本;之后再通过自动化地脚本编译,最终生成可执行文件.

本文构建的儿童画板应用程序不是一个已经固化的应用,而是一个供用户构建故事书的软件.打开新应用的可执行文件后,用户可以自己选择已有故事书,也可以新建故事书,然后进入到儿童画板的绘制界面.首先,构建一个页容器,然后进入框场景选择具体框,添加具体框并进行相应的文字或涂鸦绘制工作.图 9 展示了新应用的起始界面(左图)、Ink 场景界面(右图)和绘制完成后返回到的页场景界面(中图).在应用了 CDM 页—框的应用中,通过场景切换只改变交互对象,而应用对象(页)显示的内容并不会变,图中和图右就展示了这一关系;同时,在具体框场景进行内容编辑时,内容的变化也不会对场景交互对象产生影响.应用对象和交互对象彼此独立,在

生成新应用前在 CDM 不可见;利用 SDT 生成新的应用程序后,则把它们绑在了一起,联合发生作用,却又各自具有独立性.

Fig.9 Children's drawing board App

图 9 儿童画板应用程序

5 用户评估

开发一个新的应用程序需要经历若干子过程,而每个子过程又需要用户与设计工具的多次交互才能完成.本文从设计工具的学习成本、完成简单目标任务的用户表现以及完成项目任务的用户表现等方面,来对软件进行评估.

鉴于 SDT 设计工具"分离-融合"的设计方法,使得它不仅具有一般设计软件的界面设计能力,同时还拥有了一些界面开发软件的创造能力,因此,我们设计了两个对比实验,一个是用 SDT 和著名的设计软件 Axure 进行界面设计对比,另一个用于 SDT 和流行的 Visual Studio 2015 进行拥有复杂交互行为的产品对比.

该实验选择的设备是带有触屏功能的 Microsoft Surface Pro 和 Wacom 板,操作系统为 Windows 10;共征募了 12 名参与者,都是计算机相关领域的学生或工程师,其中计算机专业的 8 人,工业设计和多媒体设计专业的 4 人.图 10 为实验参与人员使用本文所述的软件 SDT 进行软件设计.

Fig.10 Experimental participants utilize SDT for software design

图 10 实验参与人员使用 SDT 进行软件设计

这两个评估实验的任务不同,但流程相同.在每个实验开始之前,都会让被试者观看我们提供的关于工具如何使用的视频;之后,给被试者 5min 的时间熟悉软件;最后,把预先准备好的实验任务单分发给被试者.在被试者一切准备妥当后,开始实验并计时.待被试者完成实验后,停止计时并发放问卷调查表.最后,两个实验都完成后,进行用户访谈.

5.1 用户界面设计对比实验

该实验主要从界面元素、界面布局、页面跳转几个维度出发,设计了实验任务.由于 Axure 拥有非常丰富的界面元素库,而 SDT 拥有可扩展的动态交互特性,因此实验简化了界面设计中会出现的 UI 元素类型,参与者

只需搭配使用按钮、列表、文本、图形框 4 种 UI 元素即可构建用户界面,而交互动作只选用了界面间的跳转行为.图 11 展示的两个草图用户界面,是本实验的设计目标,同时图示也为被试者清晰地展示了他们需要完成的实验任务.本实验的具体实验任务如下.

(1) 添加元素到相应界面上;

(2) 修改元素的显示文本;

(3) 更改元素的背景色或者边框颜色

(4) 为元素添加背景图片;

(5) 删除误添加的元素;

(6) 实现页面间的跳转;

(7) 调整页面布局使其与图示位置大体一致.

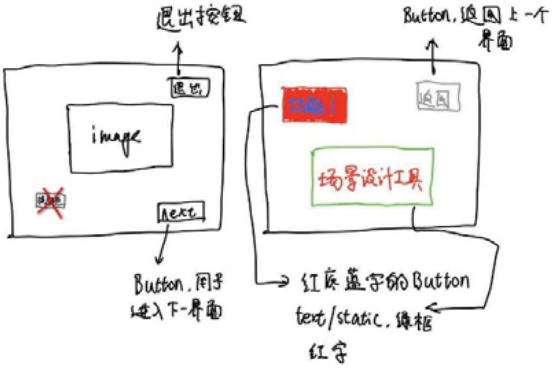


Fig.11 Experimental task sketch

图 11 实验任务草图

在实验结束后,参与者被要求填写一份调查问卷,问卷分为 7 个问题,每个问题根据用户表现分为 5 个量级,得分为 1~5.得分越高,说明参与者认为软件在该项调查中表现越好.调查问卷与统计结果见表 1.

Table 1 User interface design experiment survey statistics

表 1 用户界面设计实验调查统计表

调查内容	所用软件	
	SDT	Axure
Q1:增加元素	5	4.5
Q2:移动元素	4.8	4.9
Q3:删除元素	4.2	5
Q4:更改元素样式(背景颜色、背景图片等)	5	4.3
Q5:调整元素大小	5	5
Q6:输入文本	3.7	5
Q7:界面间跳转	5	4.4

实验结果表明:

(1) Q1~Q3 反映的是用户界面设计的前期阶段,即添加 UI 元素并将 UI 元素方式在适合的位置上的这一阶段,SDT 取得了相对于 Axure 而言较好的成绩,这说明使用草图来进行用户界面的布局设计相对于基于控件的 drag & drop 方式具有一定的优势;Q2 中 SDT 和 Axure 在移动元素问题上的用户表现基本持平;但是 Q3 中 SDT 的得分小于 Axure,根据我们对参与者的访谈,我们发现问题的直接原因在于 SDT 使用笔手势进行删除,而 Axure 可以在鼠标选中后直接使用键盘上的删除键进行操作.笔手势的绘制需要花费一定的时间,删除手势的识别也要花费一定时间而且存在识别错误的风险,这使得参与者认为 SDT 在执行删除操作时不如键盘迅捷、准确.

(2) Q4 与 Q5 中是关于元素外观表现问题,结果显示 SDT 的用户表现笔 Axure 更优,这说明使用 PGIS 交互范式的交互效率可以与使用 WIMP 交互范式的软件相等.Q4 主要针对于 UI 元素的样式、颜色等内容进行设置,参与者只需使用笔或者鼠标进行点击操作即可完成.在 Axure 中,属性的设置通过属性设置面板完成,而在 SDT 中以 gadget 的形式呈现,这说明对于笔式交互而言,gadget 可以提升程序的交互效率.

(3) Q6 是关于文字输入问题,Axure 比 SDT 取得了更好的用户表现.由于这些被试者计算机操作水平较高,都会盲打,因此,使用键盘进行文本输入的 Axure 比使用笔式手写输入的 SDT 在输入时效上取得了更佳的用户表现.

(4) Q7 是界面动作的一种简单情况,SDT 在该方面的用户表现更为优异.由于 SDT 在进行场景跳转设置时,该场景的状态迁移图直接展现在用户面前,同时触发该动作的交互对象也一目了然,很容易就能完成这一场景动作的设置;而 Axure 进行交互行为设置时,需要在不同的面板直接来回跳转,还需要额外执行切换选项卡、点击下来菜单等动作,使得其交互行为比较冗长、拖沓.

总的来说,SDT 笔式交互原语和草图的混合输入为用户提供了更多选择,使用户得到了更好的设计体验;SDT 用 Gadget 的形式进行属性设置比复杂的属性面板更加便捷、高效;SDT 进行界面跳转的步骤更加简便.但是,存在一些问题,如 SDT 的界面元素不够丰富;没有键盘的位置微调功能;删除手势复杂且存在识别错误问题;手写文本不如盲打快等.

5.2 终端应用设计对比实验

SDT 与其他用户界面设计工具相比,最大的优势在于能够使用户在设计应用程序的同时生成可直接运行的应用程序.我们称此应用程序为终端应用程序.在终端应用程序中运行期间,用户触发界面上绑定场景任务的元素后就会直接调用内部函数完成相应的操作,可以对其他元素的属性进行更改,亦可构建新的界面元素.SDT 在设计之初就考虑到了这一需求,通过第三方扩展库的字符形式供用户在设计终端程序时可以方便地通过图形化的方式进行函数调用绑定.也就是说,SDT 不仅仅具有设计用户界面的功能,而且具有一部分 IDE 的功能.因此,我们设计了第 2 个实验.该实验将通过使用 SDT 与 Visual Studio 2015 的 Winform/WPF 完成预定实验任务,进行开发效率的比较.

实验要求参与者分别使用 SDT 与 Visual Studio 进行相同功能的终端程序的生成,终端程序的样式如图 12 所示.图中的实线图形为设计阶段的可见元素,本文称其为交互对象;虚线图形为终端程序运行后,点击“新建页”、“添加框”等操作时临时创建的界面元素,本文称其为应用对象.实验要求应用对象的位置、布局需要与示意图相似或一致,编译后的终端程序中动态创建的应用对象不需要与图示完全一致,但是应用对象间的包含嵌套关系必须一致,即通过翻页操作,最外层的应用对象的所包含的所有内容跟着一起变化.此外,可以使用本地 MSDN 帮助和上网查找资料,最长时间限定为 30min.

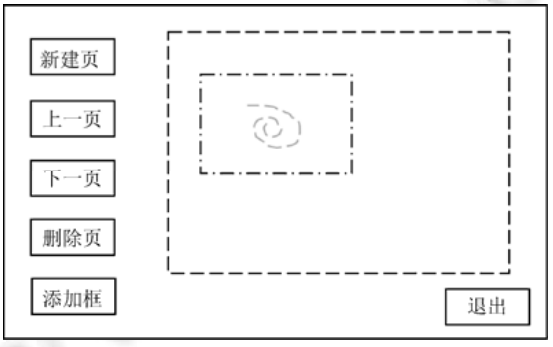


Fig.12 Terminal application schematic diagram

图 12 终端应用示意图

实验完成后,每位参与者使用两种工具完成任务所需要的时间,如图 13 所示.这 12 个人都用 SDT 完成了实验任务,平均时长为 5.1min;11 个人用 VS2015 完成了实验任务,1 个人失败,这 11 个人平均耗时 20.3min.使用 SDT 和 VS2015 完成该实验任务时,有显著差异.由此可见,SDT 这种“分离-融合”设计方法下生成的终端应用程序可以构建一些相对复杂的应用而无需编写代码,极大地提高了产品的生产效率.

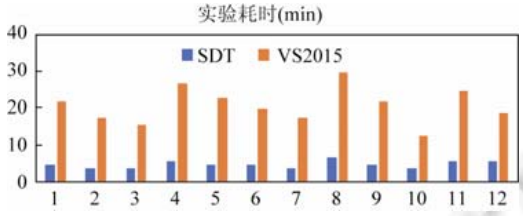


Fig.13 Terminal application design experiment time-consuming diagram

图 13 终端应用设计实验耗时图

5.3 用户反馈

在 12 个被试者做完实验的访谈过程中,8 人对 SDT 基于笔式和草图的混合输入功能给予了肯定,得到了其他软件所没有的体验;10 人对不用编程就能直接生成具有一定复杂性的软件给予了高度评价;7 人表达了对如公式在线输入识别等功能的期盼;所有参加实验的人都认为在没有键盘鼠标的平板电脑等产品上,SDT 具有更大的优势,如手写功能会比使用虚拟键盘要准确、快捷.

同时,有 2 人指出了 SDT 的删除手势识别错误的问题,有 3 人指出 SDT 在界面布局上流畅性欠佳,有 5 人指出了 SDT 软件和那商业软件在功能和界面元素丰富性上的差距.

总的来说,SDT 能够集设计和开发于一身的特性是很少软件所具有的,其“分离-融合”设计方法使得用户无需编写编码即可生成终端应用,不仅减轻了产品的开发时间,而且使得它可以为非 IT 人员所使用;同时,用它生成的终端应用软件的可用性和实用性也得到了用户的肯定.

6 结论和展望

本文基于触屏技术飞速发展但笔式交互设计工具匮乏的现状,利用场景设计方法和 PGIS 交互范式的理念,构建了同时支持传统笔式输入和智能化草图输入的场景设计工具.该工具是面向最终用户的,它不仅可以进行界面设计,还可以通过场景动作产生更多的交互行为,在经过代码自动生成和编译后可以直接生成可执行的产品.通过儿童画板实例,我们更加直观地揭示了 SDT 与 PGIS、CDM 的关系.新生成的应用程序把 PGIS 的交互对象和 CDM 的应用对象连接在一起,从而使其具有界面原型上所不具有的行为,动态添加更加丰富的内容.最后,通过界面用户对比实验,佐证了在不限定输入设备的情况下 SDT 具有与 Axure 同等的界面开发能力,在限定鼠标、键盘的情况下,SDT 在触屏设备上更加自然、高效;通过终端用户应用设计对比实验和用户反馈更凸显了 SDT“分离—融合”式设计理念所独有的创造性.

本文的场景设计工具 SDT 是为平板电脑、电子白板等触屏设备构建的设计开发工具,立足于设计工具,定位于无编码开发工具,其用户表现也得到了普遍认可;但是其界面元素相对较少,可供选择的场景动作也不够丰富,工具本身的界面也不如 Axure 和 Visual Studio 美观.因此,本文下一步的工作是扩充 PGIS 交互对象、增加适用于场景任务的函数库、丰富图片等资源库和优化界面设计.

References:

[1] Tian F. Research on post-WIMP software interface [Ph.D. Thesis]. Beijing: Graduate School of the Chinese Academy of Sciences, 2003 (in Chinese).

[2] Dai GZ, Tian F. Pen-based User Interface. 2nd ed., Hefei: Press of University of Science and Technology of China, 2014 (in Chinese).

[3] Heintz M, Law EL, Soleimani S, *et al*. Paper or pixel? Comparing paper- and tool-based participatory design approaches. In: Proc. of the Int'l Conf. on Human-Computer Interaction. 2015. 501−517.

[4] Huang F, Canny JF, Nichols J, *et al*. Swire: Sketch-based user interface retrieval. In: Proc. of the CHI 2019. 2019.

[5] Landay JA. SILK: Sketching interfaces like krazy. In: Proc. of the CHI'96. 1996. 398−399.

[6] Li Y, Guan ZW, Chen YD, *et al*. Penbuilder: Platform for the development of pen-based user interface. In: Advances in Multimodal Interfaces—ICMI 2000. 2000. 534−541.

[7] Li SH, Hsu JJ, Chang CY, *et al*. Xketch: A sketch-based prototyping tool to accelerate mobile app design process. In: Proc. of the Designing Interactive Systems (DIS 2017). 2017. 301−304.

[8] Carroll JM. Five reasons for scenario-based design. Interacting with Computers, 2000,13(1):43−60.

[9] Liang HZ, Dingel J, Diskin Z. A comparative survey of scenario-based to state-based model synthesis approaches. In: Proc. of the 2006 Int'l Workshop on Scenarios and State Machines: Models, Algorithms, and Tools (SCESM 2006). 2006.

[10] Liu J, Deng CZ, Dai GZ. Pen-based software design tool based on scenario-based method. Computer Engineering and Design, 2009, 30(8):2011−2014,2039 (in Chinese with English abstract).

[11] Segura VC, Barbosa SD, Simoes FP, *et al*. UISKEI: A sketch-based prototyping tool for defining and evaluating user interface behavior. In: Proc. of the Advanced Visual Interfaces (AVI 2012). 2012. 18–25.

[12] Szekely P. Retrospective and challenges for model-based interface development. In: Design, Specification and Verification of Interactive Systems '96. 1996. [doi: https://doi.org/10.1007/978-3-7091-7491-3_1]

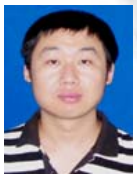
[13] Wobbrock JO, Wilson AD, Li Y, *et al*. Gestures without libraries, toolkits or training: A $1 recognizer for user interface prototypes. In: Proc. of the 20th Annual ACM Symp. on User Interface Software and Technology (UIST 2007). New York: ACM Press, 2007. 159−168. [https://doi.org/10.1145/1294211.1294238]

附中文参考文献:

[1] 田丰.Post-WIMP 软件界面研究[博士学位论文].北京:中国科学院研究生院,2003.

[2] 戴国忠,田丰.笔式用户界面.第 2 版,合肥:中国科学技术大学出版社,2014.

[10] 刘健,邓昌智,戴国忠.基于场景方法的笔式界面软件设计工具.计算机工程与设计,2009,30(8):2011−2014,2039.



仝青山(1983－),男,河北枣强人,博士生,CCF 学生会员,主要研究领域为人机交互技术,用户界面

田丰(1976－),男,博士,研究员,博士生导师,CCF 高级会员,主要研究领域为人机交互技术,虚拟现实.

张宗琦(1989－),男,硕士生,主要研究领域为人机交互.

刘杰(1981－),男,博士,高级工程师,CCF 专业会员,主要研究领域为人机交互.

黄进(1985－),男,博士,助理研究员,CCF 学生会员,主要研究领域为人机交互技术,图形图像处理.

戴国忠(1944－),男,研究员,博士生导师,CCF 杰出会员,主要研究领域为人机交互,计算机图形学.