国际儿童编程教育研究现状与行动路径

孙立会　周丹华

（天津大学 教育学院，天津 300072）

［摘要］ Logo、Tangible Programming、Scratch 等国际知名儿童编程工具各具特色。从编程形式看，Logo 语言更贴合真正的文本编程语言，对计算机程序编写有较强的启蒙作用；Tangible Programming 摆脱了单纯的计算机环境编程，使儿童在真实物理空间中参与编程活动，帮助他们接触和理解编程知识；Scratch 编程环境将编程活动形象化，通过鼠标拖拽不同功能的程序块，设计安排目标角色行为，它虽与真实编程活动有形式上的差别，但其运行机制及设计理念与编程活动所需能力一致，且操作简单，更能增强儿童编程学习的快乐体验，创造属于自己的程序项目。从能力提升角度看，Logo 编程语言对儿童认知能力的提升帮助较大，Tangible Programming 注重提升儿童排序能力，Scratch 编程环境与计算思维能力的提升有联系。国际儿童编程教育实践应用主要从学生、教师、活动形式三方面进行了大量实验研究。其中，学生维度主要关注学习兴趣、动机和自我效能感；教师维度侧重教师的课堂角色、教学经验、教学风格和技术使用舒适度；活动形式维度注重多样化的活动，如协作学习、亲子合作等。我国要在正确理解儿童编程教育本真之意的基础上，以学校为载体广泛开展实验研究；政府部门推广儿童编程教育；选择和研发适合我国儿童特点的编程工具；尽快编订与学科内容相结合的儿童编程教材，并与社会联动加快建设系统完备的具有编程思维的教师队伍。

［关键词］ 儿童编程教育；实践应用；行动路径

［中图分类号］ G434　　［文献标识码］ A　　［文章编号］ 1007-2179(2019)02-0023-13

利用适合儿童年龄发展阶段的编程工具，可帮助儿童掌握编程知识，提升理论思维能力（Portelance et al.，2014）。随着实验研究的深入，国外相继出现了 Logo、Tangible Programming、Scratch 等儿童编程工具，近半个世纪以来研究人员在教学方法、学生态度变化及能力提升等方面开展了相关研究。

一、国际儿童编程工具溯源及发展现状

（一）“海龟”绘图与 Logo 编程语言

国际第一款儿童编程工具 Logo 诞生于 1968 年，盛行于二十世纪七八十年代（Michayluk，1986），最初设计理念由麻省理工学院人工智能实验室（MIT Artificial Intelligence Lab）的西蒙·派珀特（Seymour Papert）提出。派珀特曾在瑞士日内瓦大学师从让·皮亚杰（Jean Piaget）六年，从事认知发展研究。Logo 编程工具的设计结合建构主义和认知发展阶段理论（Wilson & Sally，2001），然而依据认知发展阶段理论，儿童在前运算阶段难以掌握编程教育体现的逻辑运算及计算思维能力。派珀特提出与皮亚杰理论不同的想法，他没有对儿童的发展阶段进行划分，也不限制儿童在哪一阶段能或不能做什么（Papert，1980）。在个人计算机问世前，派珀

［收稿日期］2018-12-05　　［修回日期］2019-02-20　　［DOI 编码］10.13966/j.cnki.kfjyyj.2019.02.003
［基金项目］2018 年度教育部哲学社会科学研究重大课题攻关项目“世界主要国家教材建设研究”（18JZD017）。
［作者简介］孙立会，博士，副教授，硕士生导师，天津大学教育学院，研究方向：电子教科书、儿童编程教育（sunlihui@tju.edu.cn）；周丹华，硕士研究生，天津大学教育学院，研究方向：儿童编程教育。

特已经预见儿童将会在类似互联网的环境中,使用类似计算机的设备获取信息、辅助学习,以此提高创造力。

20 世纪 70 年代,派珀特与麻省理工学院人工智能实验室的学生们开始研究如何将孩子引入计算机世界,并将编程带进他们的物理世界(McNerney,2004)。最早版本的 Logo 并没有基于计算机环境,而是一只形似"海龟"的地面机器人。派珀特的核心思想是让儿童在物理空间学习几何概念,相比指示"海龟"做一些简单动作,不如先自身演示,然后再设计"海龟"机器人的行为(Papert,1976)。20 世纪 80 年代初,随着个人计算机的盛行,Logo 项目的推广受到前所未有的阻碍,研究人员转而开发"屏幕海龟"(McNerney,2004),即各项研究中经常使用的 Logo 编程环境的计算机版本。

Logo 语言是一种基于文本的编程语言,与自然语言非常接近(Watt,1982),它包含一个易于学习、丰富和可扩展的词汇表,反映关键的计算机概念,如局部和全局变量、命名、递归、过程和编辑等(Klahr & Carver,1988),同样秉承"海龟绘图"的方式帮助儿童学习编程知识。同时,Logo 语言的作图方式没采用坐标式,而是采用向前、后退、向左转、向右转、返回等儿童易于理解的语言和命令(Hamner & Hawley,2001),以适合儿童早期发展的身体认知,也是一种发挥儿童主体能动性的寓教于乐的教学方式。派珀特在前期主要利用 Logo 探究如何促进儿童数学能力的发展,结合课堂或个例探究 Logo 对儿童能力发展的影响及推进教师在课堂使用 Logo(Papert,1978),随后围绕 Logo 的各项研究相继展开。随着计算机技术的更新,Logo 也推出了适用于不同计算机环境及具有时代特色的进化版,如 MSWLogo、WebLogo、Easylogo、BlockLogo 等。

(二)Tangible Programming 与机器人技术

在 Logo 的基础上,Tangible Programming(有形编程)作为儿童编程工具的分支也获得迅速发展。这一概念由日本学者铃木(Suzuki)和加藤(Kato)于 1993 年提出,用来描述为儿童设计的 AlgoBlock 协作编程环境。AlgoBlock 是一种脱离计算机环境使儿童在物理空间排列程序块的编程方式,可以使儿童协作编程(Suzuki & Kato,1993)。Tangible Programming 最初也是基于派珀特早期的 Logo 编程思想。经过多年 Logo 实践教学,研究人员认识到,大多数儿童还没有准备好用传统方式编写计算机程序,就算用键盘输入 Logo 代码,至少也要到 10 ~ 14 岁才能实现。虽然"屏幕海龟"明显更实用,但它也有一定的抽象性,会给儿童的学习带来困难,因为现实世界的地板被移开,远离了儿童生活的日常环境,导致年幼的儿童难以理解。

派珀特的学生在研发"屏幕海龟"的同时也继续沿着最初的研究思路,通过将计算机编程作为创造性活动引入现实世界,弥合抽象计算和儿童学习能力间的差距(McNerney,2004)。20 世纪 70 年代中期,麻省理工学院 Logo 实验室的拉迪亚·珀尔曼(Radia Perlman)指出,阻碍儿童接触计算机编程的主要障碍不仅是语言,还有用户界面,由此珀尔曼设计了一些界面,让学龄前儿童学习"海龟编程"。按钮盒(Button Box)和老虎机(Slot Machine)是她早期设计的比较知名的儿童编程学习输入设备(Perlman,1976)。这两款设备主要利用立方块或卡片让儿童编写程序语言。此后,麻省理工学院媒体实验室的有形媒体小组(Tangible Media Group)对此进行了更深入的开发和研究(Wyeth,2002),推出有形用户界面(Tangible User Interface,简称 TUI)及利用儿童可接触的物理程序块研发儿童编程的一系列相关产品(Horn & Jacob,2007)。这种对可编程"程序块"和有形用户界面的研究直接促使创新商业产品的诞生,使有形编程逐步走上商业化之路。

与此同时,塔夫茨大学开发技术研究小组(DevTech Research Group of Tufts University)接过有形编程这面旗帜,将其与机器人技术相结合,提出有形机器人编程技术,并将这一技术应用于儿童课堂,取得令人瞩目的成绩。其中比较著名也是近期研究热点的是有形机器人(TangibleK Robotics)(Bers,2010),该项目让儿童使用一种名为 CHERP 的编程语言,即多样化环境中的创造性机器人编程活动,允许幼儿使用联锁的木块或相应的屏幕程序块进行编程,并可以在两个界面之间来回切换,用来为他们的乐高机器人设计行为动作(Sullivanl & Bers,2016)。该程序同样面向幼儿园,以开发适合的机器人编程工具与建构型课程相结合,旨在让儿童学习计算思维、机器人技术、编程原理和提高解决问题的能力(Bers et al.,2013)。TangibleK 机器人项目的设计

理念是:教孩子认识人造世界和教他们认识自然世界一样重要(Bers & Elkind,2008),通过为儿童提供更广阔的物理探索空间,让他们在创建和改造机器人程序中认识自然规律。

(三)"小精灵"动画故事与 Scratch 程序块

Scratch 作为目前儿童编程领域使用范围最广和最受欢迎的编程工具,与 Logo 和 Tangible Programming 有千丝万缕的联系。Scratch 项目始于 2003 年,是一款开源的儿童编程工具,由麻省理工学院媒体实验室的终身幼儿园小组(Lifelong Kindergarten Group)设计研发,主要适用对象是 8 ~ 16 岁儿童(Maloney et al. ,2010)。Scratch 基于 Logo 和 Squeak Etoy(Kay,2010)的思想创建,但其特征是摆脱了文本编程方式,利用鼠标拖拽不同功能的程序块编写"小精灵"的互动故事、游戏和动画,并与在线社区的其他人共享创作,使学生在简单轻松的学习环境下体会编程带来的乐趣。Scratch 还构建了"富媒体"环境,图形、声音、动画、视频、游戏和互动故事都能在编程界面有序展开(Resnick et al. ,2009)。

Scratch 创建伊始就宣传派珀特提出的"高天花板""低地板"和"宽墙"的编程教育理念(Papert,1980)。Scratch 开发团队的主设计师米切尔·雷斯尼克(Mitchel Resnick)表示,学习者应能立即轻松地创建东西(低地板),并随时间的推移保持兴趣,同时允许学生跨越多种学习风格,创建越来越复杂的项目(高天花板),从而促进学生编程认知和兴趣的发展(宽墙),在此过程中儿童编程经验和技能会随着编程工具一起成长(Resnick et al. ,2009)。换言之,Scratch 希望降低编程的上限,拓宽儿童编程的工具手段,探索创造多种编程学习路径与风格,为各种复杂概念提供生长空间,让儿童更早地开始学习编程。

2013 年,塔夫茨大学开发技术研究小组、麻省理工学院媒体实验室与 Playful 发明公司合作,推出专为 5 ~ 7 岁儿童设计的 ScratchJr。ScratchJr 是将 Logo 与 Scratch 结合的面向更低龄儿童的升级版本,主要适用对象为幼儿园到小学二年级儿童。ScratchJr 的创建解决了幼儿教育缺乏相对强大的数字创建和计算机编程技术支持问题,促进了儿童在读写和数学等成熟学术领域早期学习成果的发展,同时向儿童介绍计算机编程并加强儿童解决问题能力和基本认知技能(Flannery et al. , 2013)。ScratchJr 公开发行以来,仅在苹果应用商店就有超过 60 万次的下载,用户反响强烈(Strawhacker et al. ,2015)。如今,ScratchJr 研究团队致力于研究如何在多种学习环境中有效引入 ScratchJr 和类似技术,包括调查教育工作者如何成功地将计算机编程引入课堂,以及创建和传播有用的教学和学习资源。

有关 Scratch 的研究表明,儿童编程工具的研发逐步向低龄化发展,人们在儿童早期开展编程教育的意识越来越强,体现了 Resnick 在《终身幼儿园:通过项目、热情、同伴、游戏来培养创造性》一书中提到的与传统教育理念完全相反的观点。学校的其它时间(甚至是余生)应像幼儿园一样(Resnick,1998),要在快速变化的世界中成长,各年龄段的人都必须学会创造性地思考和行动,最好的方法就是像传统幼儿园的孩子一样,更多地关注想象、创造、游戏、分享和思考,通过给儿童提供创造性游戏的机会,让他们基于自己的热情,与同伴合作,为进入一个创造性思维比以往任何时候都更重要的世界做好充足准备(Resnick,2017)。

2018 年,麻省理工学院和 Google 发布了联合打造的最新版本 Scratch3. 0。Scratch3. 0 可供在平板计算机和手机上创建项目,还增加了与 Google 相连的语音识别技术、体感技术,并可连接外部设备,如用 Scratch 控制乐高头脑风暴积木使儿童在参与过程中体会创造程序的乐趣。

(四)小结

目前流行的儿童编程工具都来源于派珀特早期 Logo 的教育理念,其理论根基都是建构主义以及重塑了的皮亚杰认知发展阶段理论,借助适合儿童年龄发展的、可接受的技术与环境帮助他们学习掌握下一阶段的能力或顺利实现儿童两个发展阶段的平稳过渡(Feldman,2004)。其设计理念一脉相承,珀尔曼作为早期 Logo 实验室的一员,延续派珀特最初 Logo 基于物理世界的编程,使有形编程的理念在 20 世纪 90 年代再度兴起,由此电子玩具、乐高积木、机器人等与之相关的技术应运而生,尤其是塔夫茨大学开发技术研究小组的成绩瞩目。派珀特的学生雷斯尼克创造的 Scratch,同样也基于 Logo 的编程思想及有形编程中物理程序块的概念,是一款通过拖放

不同功能的程序块实现目标角色行为的图形化编程语言。

从研发团队看,国际上开展儿童编程研究的主要团队有麻省理工学院媒体实验室、塔夫茨大学开发技术研究小组,以及一些大型的儿童益智工具开发公司,如乐高集团、Playful 发明公司等。校企合作加速了儿童编程工具的研发和推广,使儿童编程走向商业化道路。从儿童编程工具版本更新的角度看,到目前为止,儿童编程工具面向的群体呈现低龄化趋势,Scratch 和有形机器人编程环境比 Logo 编程语言面向的群体年龄更低,如 ScratchJr、TangibleK Robotics,都能降低儿童编程学习的难度,并使儿童尽早接触编程知识。但这并不能作为预测今后儿童编程工具发展方向的依据,儿童编程工具在不同教学环境及学习方式中还有无限发展潜能。

二、编程工具分类比较

随着时代的发展,各类儿童编程工具不断改进,功能和特点也有不同侧重。

(一)编程形式:文本图像与实物的碰撞

Logo、Scratch、Tangible Programming 的理论根基与设计理念虽一脉相承,但各具特色。Logo 语言相对更能体现自然编程的特点。虽然 Logo 呈现的也是有着图形图像的界面,但其运行原理还是基于传统意义的计算机编程语言及过程,只是编程语言更简化,能让使用者通过给定的简化编程语言指导“海龟”作图或者进行设计创作,如开始、向前、向后、停止等。Logo 是 LISP 编程语言的变体,是探索语言和数学几何问题的一种强大而简单的工具。相对来说不受语法规则限制,因为语法规则让孩子们难以掌握编程语言(Gorman & Bourne,1983)。曾有研究者比较儿童在 Logo 和 Scratch 编程环境下的编程态度和学习效果,结果发现学习过 Logo 语言的学生对自己的编程能力更有信心(Lewis,2010)。Logo 编程语言在启蒙儿童程序语言的学习和编程信心的建立方面展现出独特优势。

Scratch 是一种更加图像化的编程环境,强调媒体操作,支持能产生兴趣共鸣的编程活动,如创建动画故事、游戏和交互式演示。Scratch 项目由固定的舞台(背景)和可移动的“小精灵”组成,每个对象都包含一组图像、声音、变量和脚本,使编程环境更可视化。它允许学生用鼠标编程:编程构造被表示为只有在“语法”合适的情况下才能组合在一起的拼图块(Maloney et al. ,2008),即儿童可以在不了解程序语言的情况下通过拖动程序块进行逻辑意义上的编程活动。所以这种环境不仅允许学生在了解编程语法前掌握编程结构,还允许关注逻辑问题(Malan & Leitner,2007),更能体现编程教育背后所蕴含的各种能力的提高。

有形编程与机器人技术倡导儿童在物理空间的编程活动,让编程活动更具体。早期的有形编程活动使儿童摆脱了操作计算机屏幕显示的虚拟对象,将空间物理程序块安排在一张桌子上,与计算机进行通信,编写操作对象的具体行为。“有形编程”指安排构建(而不是“编写”)计算机程序块活动(Mcnerney,2000),或使用可堆叠的乐高积木描述简单程序,在乐高木块中嵌入电子元件使其在系统中表达简单的行为(Wyeth & Purchase,2002)。塔夫茨大学 TangibleK Robotics 项目提出的 CHERP,使儿童在排列程序木块序列和编写程序的计算机界面之间转换,辅助年龄较小儿童尽快理解编程知识(Bers & Horn,2010)。无论何种形式的有形编程都将编程活动具体化、形象化和实物化,对低龄儿童掌握编程知识和理解编程逻辑有很大的实用价值。

综上,各种编程工具虽然在呈现形式上各有特点和优势,但 Logo 语言更贴合真正的编程语言,对计算机程序编写有更强大的启蒙作用;Scratch 将编程形象化,虽与真实编程活动有差别,但其运行机制、设计理念与编程活动所训练的能力一致,且简单易学,更能增强儿童编程学习的快乐体验,创造属于自己的程序项目;有形编程工具摆脱了单纯的计算机环境,使儿童的编程活动更有空间感,帮助大龄儿童接触和理解编程知识。各种形式的编程工具更像是个整体,互相补充,互相协作,互相促进,针对儿童编程教育的不同侧面,发挥不同作用,儿童编程工具研发推动着儿童编程教育朝着多元化方向发展。

(二)编程工具:儿童认知、排序能力、计算思维

儿童编程工具的开发和编程教育研究的落脚点是要训练思维,提升逻辑能力,促进儿童认知发展,所以各种编程工具因其自身特点,在促进儿童不同能力提升方面有区别。研究表明,Logo、Scratch、Tangible Programming 等编程工具分别对认知能力、

计算思维、排序能力等有不同程度的影响,且各种编程工具对各种能力提升的侧重点也有不同。

1. Logo 与认知能力

认知能力指人脑加工、储存和提取信息的能力,即人们对事物的构成、性能及与其它事物之间的关系、发展动力、发展方向及基本规律的把握能力,是人们成功完成活动最重要的心理条件,知觉、记忆、注意、思维和想象都被认为是认知能力。

国际上有关 Logo 语言编程的研究表明,研究者关注的重点主要是使用 Logo 编程语言与儿童认知能力的提升。派珀特及其团队创建 Logo 语言环境最初的目的是解决几何教学问题,提升学生的数学能力,但有研究发现,Logo 并没有显著提升儿童数学技能。正如克拉思诺和米特雷尔(Krasnor & Mitterer,1984)早期研究所指出的:撇开方法论不谈,几乎没有证据表明 Logo 会影响儿童的数学技能。

但与此同时,早期研究者发现 Logo 对儿童解决问题能力、计划能力、认知和元认知能力有不同程度的提升。其中认知能力是大多数研究者关注的热点。道格拉斯·H·克莱门茨(Douglas H. Clements)是儿童编程领域著名的研究者,曾比较过 Logo 环境中编程教学和计算机辅助教学对特定认知技能(分类和串行操作)、元认知技能、创造力和成就(阅读、数学和描述能力)的影响,结果发现:编程组在操作能力、元认知能力、创造力和描述能力三方面的得分明显较高,但阅读和数学成绩没有显著差异(Clements,1987)。他还曾对一年级学生进行干预并在这些儿童三年级时对其进行认知发展能力评估,结果发现,编程组儿童的认知评估分数较高,说明 Logo 编程对儿童认知能力和学习成绩有延迟影响效应(Clements,1987)。他也试图在理论层面解释 Logo 编程环境与相关认知能力的关系,提出基于斯腾伯格智力成分理论研究 Logo 环境的认知效应(包括预期这种效应的理论基础),并回顾与验证该理论基础的有效性,建议负责 Logo 环境实施和调查的研究人员将斯腾伯格的智力成分理论作为 Logo 研发的理论基础(Clements,1986)。

Logo 与认知能力提升的关系,究其原因可能是 Logo 创设的编程环境使儿童进行编程活动时需对自己的行动有预设及全景掌握,了解上一阶段或下一阶段及各步骤间的关系,以便有效地安排计划、解决问题,这正是儿童认知技能的体现。

2. Tangible Programming 与排序能力

排序是一项重要技能,也是儿童课程框架和学习评估研究的重要发现。排序是计划的组成部分,包括将对象或操作按正确的顺序排列(Zelazo et al.,1997)。比如,按照逻辑顺序复述故事,按照正确的顺序排列数字,理解一天的活动顺序等都是儿童在幼儿园和小学时必须训练和掌握的基本技能。

研究表明,有形编程在提升儿童排序能力方面作用突出。玛丽娜·内马斯基·贝斯(Marina Umaschi Bers)和伊丽莎白·R·哈萨克夫(Elizabeth R. Kazakoff)带领的塔夫茨大学技术发展研究小组 2011 ~2014 年探究计算机编程和 TangibleK 对幼儿测序能力的影响。34 名四岁半到六岁半的儿童在实验室利用 CHERP 进行三次一个半小时的实验,学习机器人的制造和编程,研究者在干预前后通过故事排序任务评估参与者的排序技能,结果发现:与编程干预前的分数相比,干预后的分数有显著提高(Kazakoff et al.,2013)。

该研究团队又研究了机器人编程对幼儿排序能力的影响,以及排序技能、班级规模、教师舒适度和技术经验之间的关系。58 名儿童被分到实验组和对照组,实验组的儿童在课堂教师指导下接受 20 小时的有形编程课程学习,研究者在干预前和干预后使用图片故事排序任务评估所有参与者的测序技能。结果发现,小组作业与测试结果之间存在显著的交互作用,即编程组的排序能力提高更明显(Kazakoff & Bers,2012)。

他们还探究了基于课堂的密集型机器人和编程研讨小组对幼儿园儿童排序能力的影响。该研究以纽约市一家 STEM 幼儿园为实验对象,以编程研讨小组的形式对 27 名儿童进行机器人有形编程训练,并在干预前后要求这些儿童完成图片故事排序任务。结果表明:与测试前相比,参加为期一周机器人有形编程研讨小组的儿童在测试后的排序得分提高显著(Kazakoff et al.,2013)。

有形机器人编程有助于提高儿童排序能力的原因在于:一,有形编程是让儿童按预定的目标在物理空间正确地排列程序块,以使目标对象产生预设行为,主要训练形式是顺序排列,掌握程序块的正确顺序才能设计机器人或目标对象的具体行为;二,有形

编程,尤其是机器人编程大多面向幼儿园儿童,排序能力是这个阶段的儿童日常生活和学校学习中重点培养的能力,有形编程能使儿童将排序能力具体化,帮助他们更好地掌握和提升此能力。

3. Scratch 与计算思维能力

进入 21 世纪,人类生活在各种软件驱动的数字生态系统中,计算思维正成为一种不可或缺的技能。周以真教授曾指出:"计算思维是每个人的基本技能,除了阅读、写作和算术外,我们还应该在每个孩子的分析能力中加入计算思维能力"。她还将计算思维定义为一种利用计算机科学基本概念解决问题、设计系统和理解人类行为的方法(Wing,2015)。

研究者经常将 Scratch 与计算思维能力进行不同形式的捆绑研究,如利用 Scratch 编程工具学习计算思维的概念以促进计算思维能力的提升。在教学形式上,曾有研究者提出一种新的评估儿童计算思维的方法,让三所学校的二年级学生使用 ScratchJr 学习基础计算思维概念,并将所学知识应用于创建拼图动画、故事和游戏,研究者用 iPad 对儿童进行视频采访,通过采访儿童对自己所设计作品的介绍,研究者清晰地了解儿童使用 ScratchJr 创作作品过程中的思维过程,此种评估计算思维的方式比传统量表式评估更有效(Portelance & Bers,2015);还有研究者通过远程教学的方式利用 Scratch 软件帮助儿童学习计算思维概念,提升解决问题的能力(Marcelino et al. ,2018);有研究者设计了 Dr. Scratch,检查儿童的编程过程是否正确及其能否发展和评估儿童的计算思维技能(Morenoleon et al. ,2015)。总之,Scratch 虽是图形化编程语言,但它还是基于计算机环境利用鼠标和键盘进行编程,需要儿童掌握一定的计算机概念,适应利用计算机进行编程;在拖拽不同功能程序块设计编程块顺序时,需要对图像化的全局进行系统的全面分析,理解角色的行为逻辑顺序,以实现角色的预设目标,也许这正体现了计算思维的概念内涵。

本文根据国际上大多数研究者的研究方向和重点,就编程工具对提高儿童不同能力的作用进行基本的分类总结,并不能说明一种编程工具只针对性提升儿童单一能力。佩亚等曾探究 Logo 编程与规划能力之间的关系,塔夫茨大学贝斯(Bers,2010;Bers et al. ,2014)关注 TangibleK 机器人课程与早期儿童课堂应用计算思维能力的关系。基于上述分析,我们设想造成这种现象的原因可能在于各种编程工具的形式特点、创设的编程环境界面及运行机制不同,在训练儿童不同能力方面各有千秋,且各种能力不是独立的,认知能力、计算思维能力、排序能力在形式及表现效果方面都有交叉,这也是国际儿童编程研究百花齐放的一个重要原因。

三、实践应用

研究者在学校及实验室开展了系列实践,从多形式多角度探讨如何将儿童编程工具引入课堂教学及如何在课堂实践中开展适合儿童发展的编程,以促进儿童获得编程知识和提升理论能力。

(一)学生的学习兴趣、动机和自我效能感

学生通常认为编程是乏味而艰难的,学习效果不理想(Papadakis et al. ,2014)。儿童编程工具的出现是为了让编程更容易被儿童接受,且特定方式的编程工具能提高学习编程的兴趣和增强自我效能感。部分研究者在关注儿童学到多少认知概念的同时也关注儿童对编程知识学习的兴趣及对编程活动的自我效能感。

威尔逊等指出,理想的教育系统要像帮助学习者如何设计程序一样关注他们的情感状态。例如,为了测量儿童在课上使用 Scratch 的感兴趣程度或者是否觉得无聊沮丧,威尔逊等在每节课后都要求儿童用三个小表情(悲伤/中立/微笑)填写简短的心情日志表,说出他们课上做了什么,以及他们对学习的感受。结果表明,儿童在编程认知上没有显著进步,但最大的好处是学习编程能带来很多乐趣,使学习成为积极的体验,与以往看起来挫败和焦虑的学习形成鲜明对比。因此,威尔逊等指出,情感对学习很重要,而不仅仅是作为辅助,学习没有动力就很难进步,且需要由积极的因素激发和维持(Wilson & Moffat,2010)。

菲利兹等也曾做过类似探究,发现 Scratch 对小学生的问题解决能力没有带来明显影响,但"对自己解决问题的能力有信心"这一因素的平均值有所增长,但无显著差异,且大多数学生认为 Scratch 平台易于使用(Kalelioglu & Gulbahar,2013),这说明儿童使用 Scratch 学习编程没有觉得困难而产生无能感,而是增强了编程的信心。还有研究者将

Scratch 应用于中学生信息技术课程学习,结果虽无显著差异,但 Scratch 能促进中学生更高级的学习,使学生学习新课题需要的时间变少,降低了学生课程学习的困难,且学生表现出更高的学习动机和自我效能感(Armoni et al. ,2015)。

西班牙国立远程教育大学的研究者曾对五所小学的 107 名五年级和六年级学生进行为期两年的追踪研究,在科学和艺术学科中整合编码和视觉程序编程,评估课堂练习中 Scratch 编程语言的使用情况,分析学生学习成果和态度。量表测试和基于对学生与观察者的描述结果表明:使用可视化编程可以为学生提供乐趣、动力、热情和承诺,在干预过程中进行的基于项目的学习使主动学习成为可能。通过这种教学方法,学生的动机、乐趣、承诺和热情得到了体现,他们完全赞成这种教学设计方法。研究者建议教育部门应在小学五六年级实施编程教学,编程教育应在所有领域交叉实施,特别是在社会科学和艺术领域,且要考虑学科领域视觉内容的特点,从积极角度创建丰富多彩、动态和鼓励性的项目(Sáez-López et al. ,2016)。

正如儿童编程教育理念所倡导的,儿童编程教育不是为了将晦涩的编程语言传递给年幼的儿童,而是让他们体验编程的过程,建构属于自己的项目,享受编程带来的快乐。结果也证明,Scratch 编程环境能够给学生积极的编程体验,增强学习的动机和自我效能感。关注儿童编程学习过程的情感因素也许更能体悟儿童编程教育的真谛。

(二)教师的角色、经验、教学风格、技术使用舒适度

儿童编程教育的核心是丰富学生体验,但编程教育实践应用能在课堂有效开展,教师的作用不可忽视。国际上早期关于儿童编程教育课堂应用的研究案例大部分是研究人员充当教师的角色,指导教学活动的开展。但目前研究者更关注在真实的课堂环境中,由真正的任课教师使用编程工具指导教学活动和课堂管理,研究人员只作为背后的设计者和辅助者,这也是当前及未来的趋势。因此,教师对新技术的态度、教学经验、教学风格及技术使用的舒适度等都是课堂实践应考虑的内容。

塔夫茨大学技术开发研究小组的研究者将教师视为儿童“强大的”想法及课堂管理的设计者,指导儿童使用乐高头脑风暴积木与 ROBOLAB 设计属于自己的机器人程序,并通过介绍四个学习故事展示教师应如何识别儿童强大的想法,使他们能设计自己的项目,同时考虑儿童能力和发展的差异,为年幼学生提供参与设计体验学习的机会(Bers,2002)。

费萨基斯等对 5 ~ 6 岁儿童使用计算机编程解决问题的维度进行了探索性案例研究。教师通过简短的介绍性体验游戏,让孩子们在 Logo 环境的交互式白板上,参与解决一系列类似的计算机编程问题。该游戏作为幼儿园结构化学习活动的一部分在全班进行。通过对干预录像的观察,以及对教师访谈和研究者笔记的分析,该研究发现:儿童喜欢参与这种编程学习,并能借此机会发展数学概念、问题解决能力和社交技能;教师积极的态度以及作为协调者和促进者的角色对学生课堂编程活动的有效开展起重要作用。教师在实验结束后的采访中表示,今后愿意让学生参与类似活动,认为这种模式加强了儿童的协作和积极参与,鼓励学生间广泛对话与交流。可见,教师在编程学习活动中的作用至关重要,能鼓励并支持孩子克服困难,控制和协调教学活动中出现的问题(Fessakis et al. ,2013)。

斯特拉沃克等探究了教师独特的教学风格与学生探索和保留编程内容能力间的关系。这项混合设计研究针对美国六所学校的 6 名教师和 222 名幼儿园到小学二年级的学生收集定量和定性数据,这些教师和学生参与了教师教学风格(专家型、正式权威型、个人型、促进者型和全权代表型)与学生学习成果关系的调查。所有参与者至少参加两节课(最多七节课),且教师报告他们的课堂结构、课程计划、教学风格和技术使用的舒适度,然后采用 ScratchJr 评价工具(ScratchJr Solve It)进行评估,以获取学生对编程理解的信息,分析学习成果的趋势。研究结果表明,所有学生都成功地理解了 ScratchJr 基础编程知识;课程规划具有灵活性;对学生需求能作出反应;拥有技术内容专业知识及注重培养学生独立思考能力的教育工作者,即专家型和促进者型教学风格的教育工作者;学生在编程中有更好的表现。实验结束后对教师的访谈结果表明,无论教学风格如何,大多数教师认为 ScratchJr 开放式的设计是成功的关键,该编程环境能给予学生承担风险的勇气和信心、犯错误的宽容、可接受的调试和解决问

题的过程,以及追求实现目标的渴望(Strawhacker et al. ,2017)。可见,教师对 ScratchJr 编程环境的积极评价有助于其在教学中的有效应用。

塔夫茨大学哈萨克夫和贝斯研究了机器人编程对儿童早期编程排序能力的影响,以及排序能力、班级规模、教师的舒适度和技术使用经验之间的关系。这个测验在两所学校进行,两所学校的班级规模、教师对技术的熟练程度不同,一位教师已多年使用技术进行教学,另一位教师没有使用技术的经验。根据学校环境的不同,研究者将儿童分成实验组和对照组。实验组的儿童在教师指导下,接受 20 小时的 TangibleK 编程课程。研究者在干预前后通过图片故事测序任务评估 54 名参与者的测序技能,并使用重复测量方法进行分析。结果表明,不同学校的测试结果没有显著差异,学校内部不同组别之间测试结果有显著性差异。与此同时,研究者还讨论了班级规模、教师经验、教师舒适水平与技术等因素的影响作用,发现对较小规模班级进行实验干预更有效,教师经验和教师技术使用的舒适度没有对实验结果产生显著性差异,只是在平均分上有变化(Kazakoff & Bers,2012)。

以上研究案例表明教师的重要性,包括教师对技术的积极态度和体验能增加他们在课堂环境中使用编程工具的次数,加快编程工具进入课堂的速度;准确的教师角色定位和恰当的教学风格对儿童课堂编程教育活动的有效开展和课堂管理有一定帮助,教师以协调者和促进者的角色指导编程课堂活动,有助于儿童更快更主动地掌握编程知识;教师编程工具技术经验对儿童编程知识概念的有效获取的影响不显著,说明儿童编程工具进入课堂对教师的技术要求不高,教师通过适当的培训就可选择适当的编程环境进行教学。

(三)教学活动开展形式

在将儿童编程教育引入课堂的过程中,研究人员也从教学活动开展方式的角度进行了大量研究,为儿童编程教育工具进入课堂和课堂编程活动的有效开展提供了外部支持,也对儿童编程教学活动在课堂多样化地展开有一定的指导意义。

葡萄牙科英布拉大学(University of Coimbra)的研究人员在其远程教育项目中开发并运行了一门远程教育课程,该课程专门为小学教师设计,以 Moodle 学习管理系统为课程载体学习计算思维概念和 Scratch 软件操作。首次网上教学经验和结果表明:学员可以学习计算思维概念和 Scratch 编程,可以利用这种教与学的模式为课堂实践开发有用产品(Marcelino et al. ,2018),也为在课堂中进行 Scratch 编程课程提供了一种尝试形式。

塔夫茨大学的研究者采用视频访谈的方式评估儿童计算思维学习。他们选取三所学校的二年级学生,让其使用 ScratchJr 学习计算思维相关概念,然后将所学知识用于创建动画、故事和游戏,进而用 iPad 摄像头对彼此进行视频采访,在参与 ScratchJr 项目的第 4、8 和 12 天之后的每一天,学生成对地会面,使用他们最近创建的 ScratchJr“动画类型”项目视频访谈。在每次采访中,一名学生(采访者)在拍摄视频时打开 iPad 摄像头应用程序,另一名学生(主持人)打开 ScratchJr 应用程序,向摄像机展示她的项目。视频访谈提供了一种替代方法,可以捕捉到丰富的数据,也可以在真实情境中理解儿童的思维过程(Portelance & Bers,2015)。同伴视频访谈作为一种课堂活动和评价方法的细化,对于以儿童为研究对象的计算机科学教育教学研究人员来说是值得借鉴的尝试。

此外,指导协作和小组学习也是在课堂中展开编程教育活动的探究方式之一,有研究者调查了指导协作能否提高小学生在 MSWLogo 中编程学习的成绩。四年级学生被随机分配到个人学习组、自由协作组或指导协作组。所有学生都被要求按照分析、设计、编码、调试和反思等问题解决的步骤进行学习,但只有指导协作小组的学生被要求在每个步骤中执行所有指定的活动。学生考试成绩统计分析结果显示:指导协作小组的表现明显优于其他两组,三组的低成就者之间的差异尤为显著,表明他们从指导合作中获益最大。研究还发现,指导下的协作促进了学生之间有意义的讨论,有助于他们更好地理解编程概念和掌握解决问题的技巧,从而提高考试成绩(Lin et al. ,2007)。

亲子合作和互动也是研究者关注的方向之一。新技术正慢慢进入幼儿课堂,但是很多家庭在儿童进入学校之前就已经接触到了这些技术,因此许多家长成为孩子的第一批技术素养教师,研究者利用 5 个周末教 17 对父母和 20 个孩子使用机器人编程

乐高积木创建有意义的项目,包括编程和构建组件。参与者分为亲子组和儿童独立编程小组。研究结果表明:亲子合作小组的最终评价项目显著优于儿童独立编程小组,其他项没有显著差异。研究者认为,亲子组的儿童没有比独立小组儿童表现更突出的原因是父母可能更专注于自己的创作或者没有将自己的教学水平调整到适合孩子可以理解的程度,即父母没有考虑孩子的水平和程度(Beals & Bers,2006)。贝斯2007年开展了类似研究,在为期5周的研讨会中,4~7岁的儿童及其父母一起在一个多代机器人实践社区中构建和规划有个人意义的机器人项目,研讨会的目标是教父母和孩子们机器人技术中涉及的机械和编程知识,并引导他们学习技术,该项目提供了父母和孩子一起学习的不同方式,并提供了如何开发教育干预措施的见解,这些干预措施将在机器人和新技术等新知识和技能领域指导父母和孩子们的学习(Bers,2007)。

综上,国际儿童编程教育课堂应用实践研究呈现丰富多彩之态,教师和学生是课堂不可或缺的两个主体。研究者在将编程教育引入儿童课堂时不仅要关注儿童能否有效地掌握编程知识,而且也将目光转向儿童编程学习的情感过程,儿童学习兴趣、学习动机和自我效能感的提升是编程教育在课堂应用的前提和基础,并且也更能体现编程教育的理念精髓,即降低编程学习难度,使儿童去体验,享受学习的乐趣,提高学习动机,增强信心;教师作为教学活动的协调者和促进者在儿童编程课堂中发挥着重要的作用,教师对新技术的适应和对编程工具的积极评价能够促进编程活动尽快融入课堂,根据学生特点在不同教学情境中采用适合学生的教学风格,有效地组织和管理课堂活动,这些是编程教学活动有序开展的基础;同时,课堂编程活动的引入和开展需要更加多样化的形式,研究者设计多种技术支持和互动方式,给儿童课堂编程教学以有力的外部辅助,并增强其发展活力。

四、启示与建议

随着STEAM教育被写进国策,STEAM教育概念下的儿童编程教育正在我国经历从“非刚需”到“刚需”的转变。但我国儿童编程教育相较于国外起步较晚,还没有形成相对系统且成熟的编程教育体系,在编程理念普及、政策支持、编程工具、编程教育教学活动体系、儿童编程教材、师资配备等方面还有很多不足,国际儿童编程教育丰富的研究和实践成果对我国儿童编程教育的开展有一定启示作用和借鉴意义。

第一,要正确理解儿童编程教育的本真之意。儿童编程教育体现的是“用编程学”而不是“学编程”的思想,切勿受社会选拔等因素的影响将儿童编程教育作为计算机编程教育的“少儿化”,这不仅将影响儿童认知能力的发展,也会遏止儿童对计算机研究的好奇心。正如国际儿童编程教育发起者派珀特所倡导的,儿童编程教育工具的研发是为了降低儿童编程的门槛,为儿童创设广阔而自由的发展空间,使他们创造属于自己的项目,提高学习信心,体验编程学习带来的乐趣;在国际编程教育的课堂实践应用研究中,研究者也更多关注学生编程学习过程中的情感体验,学习动机、兴趣和自我效能感的提升更能体现儿童编程教育理念的真谛;儿童编程教育作为一种辅助儿童提高认知和思维能力的学习方式,最重要的是教授如何思考而不是思考什么。

儿童编程教育的开展和普及不应作为学业选拔的手段,而应作为提升儿童能力的方式,使他们能够掌握编程教育背后所蕴藏的能力本质,对其日后的学习生活产生指导作用。因此,研究者将研究重心放在不同编程工具对儿童理论思维能力的提升,认知能力、排序能力、计算思维能力都是儿童编程活动应重点训练且对儿童学习生活有深远意义的技能。我国教育工作者在推广普及儿童编程教育时,应精准把握儿童编程教育理念精髓,以更加开阔和长远的视角看待儿童编程教育,注重编程学习过程中儿童的积极体验,以提升兴趣、通过训练提升理论能力为出发点和落脚点,使儿童编程教育教学活动能够真正有效地开展。

第二,儿童编程教育大规模开展需要以学校为载体,尤其是公立学校,这离不开政府部门硬性政策和规定的支持。美国在儿童编程教育方面的开展和推广值得我们借鉴,2015年美国政府计划在10年内普及中小学生编程教育,政府投资40亿美元开展儿童编程教育;2017年特朗普签署备忘录,拨款2亿美元支持STEAM教育,重点支持计算机科学;相关政府部门鼓励编程技术工具的研发,并支持在学

校开展教育和研究活动,使美国成为儿童编程教育的发源地和领军者。日本文部科学省已将小学编程教育写进新一轮《小学学习指导纲要》,并于2018年3月出台了小学编程教育的实施步骤(第一版),小学编程教育作为日本小学课程改革必修课推进过程,将在2020年全面实施(森秀樹等,2001)。英国政府规定儿童5岁进入小学起就必须学习如何编写程序,一直到16岁完成中学学业。韩国也于2018年将编程教育纳入小学正规必修课程。

儿童编程教育研究活动大多是在学校中开展的,尤其是公立学校,政府部门强有力的政策支持使儿童编程教育顺利进入学校、走进课堂,让绝大多数儿童能够接触到先进的科学知识和技术环境。研究者在试验后提倡学校尽快开展儿童编程教育。但目前我国的儿童编程教育大多由一些教育科技公司举办,或以竞赛的形式作为学生学业加分的工具,这显然不利于儿童编程教育的发展,并且也违背了儿童编程教育的初衷。因此,我国政府也应尽快出台相关政策,加快儿童编程教育作为学校必修课程的进程,倡导惠及全体学生的更加公平的儿童编程教育,让学生都能体验到编程教育的意义和内涵。

第三,儿童编程教育的开展离不开编程技术工具的支持。面对众多儿童编程工具,教师要根据不同编程工具的形式特点和培训能力侧重点,并结合我国儿童的编程认知能力、技术接受度等开展教学活动。同时,我国也应该积极开展有本国特色的儿童编程工具研发,这需要教育科研机构和儿童编程科技公司共同努力。

目前,我国儿童编程工具的研发还主要集中在一些教育科技公司,近两年来越来越多的儿童编程项目和儿童编程公司成立,也为我国儿童编程教育市场注入新活力,但不容忽视的是绝大部分公司还处于初始发展阶段,融资不足、发展欠缺的问题十分明显,这同样需要政策和市场的支持与包容,也希望我国儿童编程教育工具研发机构和部门能够借鉴国际经验,开发更具中国特色的、适合儿童发展和更系统成熟的儿童编程工具。同时,儿童编程教育科技公司也应加强与高等院校、科研机构和小学、幼儿园的联系,加速推动儿童编程教育的研发、推广和普及。国际研究也印证了这一点,乐高集团一直以来都是Scratch编程环境和机器人编程项目最忠实的合作者和支持者,儿童编程工具与乐高积木结合形成了独具特色的品牌效应,同时也推动儿童编程工具研发更新以及市场推广。因此,我国儿童编程教育的开展也可以参考这一模式,以高校和科研机构为技术依托,企业为市场导向,学校为实践基地和信息反馈来源,三位一体互补融合,使各种技术和信息及时传达和调整,加速我国儿童编程教育工具研发及课堂实践应用。

第四,我国虽然在逐渐提高对儿童编程的重视,并且儿童编程教育已经在部分地区的课堂展开,但是儿童编程教育的课堂应用依然比较落后,没有形成系统的课程和教学体系、教材以及师资队伍。

Scratch虽然是当前我国市面上历史最悠久的编程工具,但它仅是一款工具,在我国还没有形成系统的课程和学习方案。这就需要儿童编程教育研发人员和课堂教师合作,针对不同编程工具特点,开发适合我国儿童课堂学习的系统且成熟的课程及教学体系。国际上有关各种编程工具的研发以及应用的进展同样可以给我们借鉴和启示,在设计课程时要同时注意学生的认知提升以及情感认同,培养儿童编程学习兴趣,同时也要考虑教师对编程教学引入课堂和活动开展的影响,采用不同形式的编程教学方式以及评价方式,如远程教学方式、同伴视频采访评价、协作学习、亲子合作等。

同时,我国也需要尽快编订适合不同年龄发展阶段和学科特点的儿童编程教材,同时要考虑不同学科所呈现的视觉内容特点,结合不同单元和章节重点内容将编程知识融入其中,创建丰富多彩的、动态的和鼓励性积极体验项目,使儿童在编程学习过程中掌握课程知识,让教材真正体现用编程学的理念。

师资匮乏和教师编程教育课堂经验不足也是制约我国儿童编程教育发展的一大问题。儿童编程领域缺少既懂得编程专业知识又有教育专业背景的师资团队,这就需要形成健全的教师培训体系,使他们能够精准地掌握儿童编程教育的理念精髓与技术操作,结合本学科特点及研究人员的外部支持,逐渐形成具有学科特色的编程教育课程体系。同时,教师是否认同和适应儿童编程工具在课堂中的使用也是影响编程工具进课堂的关键因素,教师对编程工具的积极评价能够加速编程教育在学校的推广,儿童

编程教育作为一种能力教育,更加要求教师注重捕捉和鼓励学生在课堂上的“强大”想法和灵感,以协调者和促进者的身份开展编程教学活动和课堂管理,不可越俎代庖,更不能敷衍应付,要以学生为活动主体促进编程活动在我国儿童课堂的有效开展。

[参考文献]

[1] Armoni,M.,Meerbaumsalant,O.,& Benari,M.(2015). From Scratch to “Real” programming[J]. ACM Transactions on Computing Education,14(4):25.

[2] Beals,L.,& Bers,M. U.(2006). Robotic technologies: When parents put their learning ahead of their child's[J]. The Journal of Interactive Learning Research,17(4):341-366.

[3] Bers,M. U.(2007). Project interActions: A multigenerational robotic learning environment[J]. Journal of Science Education and Technology,16(6):537-552.

[4] Bers,M. U.(2010). The TangibleK robotics program: Applied computational thinking for young children[J]. Early Childhood Research and Practice,12(2):1-20.

[5] Bers,M. U.,& Elkind,D.(2008). Blocks to robots: Learning with technology in the early childhood classroom[M]. Teacher's College Press.

[6] Bers,M. U.,Flannery,L. P.,Kazakoff,E. R.,& Sullivan,A.(2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum[J]. Computers & Education,72(1):145-157.

[7] Bers,M. U.,& Horn M,S.(2010). Tangible programming in early childhood: Revisiting developmental assumptions through new technologies[M]. Information Age Publishing.

[8] Bers,M. U.,Ponte,I.,Juelich,K.,Viera,A.,& Schenker,J.(2002). Teachers as designers: Integrating robotics into early childhood education[J]. Information Technology in Childhood Education,9(1):123-145.

[9] Bers,M. U.,Seddighin,S.,& Sullivan,A.(2013). Ready for robotics: Bringing together the T and E of STEM in early childhood teacher education[J]. The Journal of Technology and Teacher Education,21(3):355-377.

[10] Clements,D. H.(1986). Effects of Logo and CAI environments on cognition and creativity[J]. Journal of Educational Psychology,78(4):309-318.

[11] Clements,D. H.(1987). Longitudinal study of the effects of Logo programming on cognitive abilities and achievement[J]. Journal of Educational Computing Research,3(1):73-94.

[12] Clements,D. H.(1986). Logo and cognition: A theoretical foundation[J]. Computers in Human Behavior,2(2):95-110.

[13] Feldman,D. H.(2004). Piaget's stages: The unfinished symphony of cognitive development[J]. New Ideas in Psychology,22(3):175-231.

[14] Fessakis,G.,Gouli,E.,& Mavroudi,E.(2013). Problem solving by 5 - 6 years old kindergarten children in a computer programming environment: A case study[J]. Computers & Education,63(11):87-97.

[15] Flannery,L. P.,Silverman,B.,Kazakoff,E. R.,Bers,M. U.,& Resnick,M.(2013). Designing ScratchJr: Support for early childhood learning through computer programming[C]. In proceedings of Interaction Design and Children:1-10.

[16] Gorman,H.,& Bourne,L. E.(1983). Learning to think by learning Logo: Rule learning in third grade computer programmers[J]. Bulletin of the Psychonomic Society,21(3):165-167.

[17] Hamner,B.,& Hawley,T.(2001). Logo as a foundation to elementary education[J]. Education,108(4):463-468.

[18] Horn,M. S.,& Jacob,R. J. K.(2007). Designing tangible programming languages for classroom use[C]. In proceedings of the 1st Annual International Conference on Tangible and Embedded Interaction:159-162.

[19] Ishii,H.,& Ullmer,B.(1997). Tangible bits: Towards seamless interfaces between people,bits and atoms[C]. In proceedings of CHI'97:1-8.

[20] Kalelioglu,F.,& Gulbahar,Y.(2013). The effects of teaching programming via Scratch on problem solving skills: A discussion from learners´perspective[J]. Informatics in Education,13(1):33-50.

[21] Kay,A.(2010). Squeak Etoys,children,and learning.[EB/OL].[2018-12-01]. http://www.squeakland.org/resources/articles/(accessed 6/10).

[22] Kazakoff,E. R.,& Bers,M. U.(2012). Programming in a robotics context in the kindergarten classroom: The impact on sequencing skills[J]. Journal of Educational Multimedia and Hypermedia,21(4):371-391.

[23] Kazakoff,E. R.,Sullivan,A.,& Bers,M. U.(2013). The effect of a classroom-based intensive robotics and programming workshop on sequencing ability in early childhood[J]. Early Childhood Education Journal,41(4):245-255.

[24] Klahr,D.,& Carver,S. M.(1988). Cognitive objectives in a Logo debugging curriculum: Instruction,learning,and transfer[J]. Cognitive Psychology,20(3):362-404.

[25] Krasnor,L.,& Mitterer,J.(1984). Logo and the development of general problem solving skills[J]. The Alberta Journal of Educational Research,30(2):133-144.

[26] Lewis,C. M.(2010). How programming environment shapes perception,learning and goals: Logo vs. Scratch[C]. In proceedings of Technical Symposium on Computer Science Education:346-350.

[27] Lin,J. M.,Li,Y. L.,Ho,R.,& Li,C. C.(2007). Effects of guided collaboration on sixth graders' performance in Logo programming[C]. In proceedings of the 37th Annual frontiers in education conference:11-16.

[28] Malan,D. J.,& Leitner,H. H.(2007). Scratch for budding

computer scientists[J]. Technical Symposium on Computer Science Education,39 (1):223-227.

[29] Maloney,J. ,Resnick,M. ,Rusk,N. ,Sliverman,B. ,& Eastmond,E. (2010). The Scratch programming language and environment [J]. ACM Transactions on Computing Education,10(4):16.

[30] Maloney,J. H. ,Peppler,K. A. ,Kafai,Y. B. ,Resnick,M. , & Rusk,N. (2008). Programming by choice: Urban youth learning programming with Scratch[J]. Technical Symposium on Computer Science Education,40(1):367-371.

[31] Marcelino, M. J. , Pessoa, J. , Vieira, C. , & Salvador, T. (2018). Learning computational thinking and Scratch at distance[J]. Computers in Human Behavior,80(9):470-477.

[32] McNerney,T. S. (2004). From turtles to tangible programming bricks: Explorations in physical language design[J]. Personal and Ubiquitous Computing,8(5):326-337.

[33] Mcnerney,T. S. (2000). Tangible programming bricks: An approach to making programming accessible to everyone[M]. The MIT Press.

[34] Michayluk,J. O. (1986). Logo: More than a decade later [J]. British Journal of Educational Technology,17(1):35-41.

[35] Morenoleon,J. ,Robles,G. ,& Romangonzalez,M. (2015). Dr. Scratch: Automatic analysis of Scratch projects to assess and foster computational thinking[J]. Revista de Educación a Distancia,46(9):1-23.

[36] Papadakis,S. ,Kalogiannakis,M. ,Orfanakis,A. ,& Zaranis, N. (2014). Novice programming environments. Scratch & App inventor: A first comparison[C]. In proceedings of the 2014 Workshop on Interaction Design in Educational Environments:346-350.

[37] Papert,S. (1976). A case study of a young child doing turtle graphics in Logo[C]. In proceedings of National Computer Conference: 1049-1056.

[38] Papert, S. (1980). Mindstorms: Children, computers, and powerful ideas[M]. New York:Basic Books.

[39] Papert,S. (1978). Interim report of the Logo project in the brookline public schools:An assessment and documentation of a children's computer laboratory[EB/OL]. [2018-12-01]. https://eric. ed. gov/?id = ED207799.

[40] Pea,R. D. ,& Kurland,D. M. (1984). Logo programming and the development of planning skills[EB/OL]. [2018-12-03]. https://eric. ed. gov/? q = ED249930&id = ED249930.

[41] Perlman,R. (1976). Using computer technology to provide a creative learning environment for preschool children[M]. MIT Artificial Intelligence Laboratory Publications.

[42] Portelance,D. J. ,& Bers,M. U. (2015). Code and tell: Assessing young children's learning of computational thinking using peer video interviews with ScratchJr[C]. In proceedings of the 14th International Conference on Interaction Design and Children:271-274.

[43] Portelance, M. J. , Strawhacker, A. L. , & Bers, M. U. (2016). Constructing the ScratchJr programming language in the early childhood classroom[J]. International Journal of Technology and Design Education,26(4): 489-504.

[44] Resnick,M. (2017). Lifelong kindergarten: Cultivating creativity through projects,passion,peersand play[M]. The MIT Press.

[45] Resnick,M. (1998). Technologies for lifelong kindergarten [J]. Educational Technology Research and Development,46(4):43-55.

[46] Resnick,M. ,Maloney,J. ,Monroyhernandez,A. ,Rusk,N. , Eastmond,E. ,Brennan,K. ,Millner,A. ,Rosenbaum,E. ,Silver,J. S. , Silverman,B. ,& Silverman,Y. B. (2009). Scratch: Programming for all[J]. Communications of The ACM,52(11):60-67.

[47] Sáez-López,J. M. ,Román-González,M. ,& Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using "Scratch" in five schools[J]. Computers & Education,97(3):129-141.

[48] 森秀樹,杉澤学(2001). Scratchを用いた小学校プログラミング授業の実践[J].日本教育工学会論文誌,34(4):5-10.

[49] Strawhacker,A. L. ,Lee,M. ,& Bers,M. U. (2017). Teaching tools, teachers' rules: Exploring the impact of teaching styles on young children's programming knowledge in ScratchJr[J]. International Journal of Technology and Design Education,28 (2):347-376.

[50] Strawhacker,A. ,Lee,M. ,Caine,C. ,& Bers,M. U. (2015). ScratchJr demo: A coding language for kindergarten[C]. In proceedings of Interaction Design and Children:414-417.

[51] Sullivanl,A. ,& Bers,M. U. (2016). Robotics in the early childhood classroom: Learning outcomes from an 8-week robotics curriculum in prekindergarten through second grade[J]. International Journal of Technology and Design Education,26(1):3-20.

[52] Suzuki,H. ,& Kato,H. (1993). AlgoBlock: A tangible programming language, a tool for collaborative learning [C]. In proceedings of the 4th European Logo conference:1-10.

[53] Watt, M. (1982). What is Logo? [J]. Creative Computing, 8(10):12-17.

[54] Wing,J. M. (2006). Computational thinking[J]. Communications of The ACM,49(3):33-35.

[55] Wilson,A. ,& Moffat,D. C. (2010). Evaluating Scratch to introduce younger school children to programming[C]. In proceedings of the 22nd Annual Psychology of Programming Interest Group:1-12.

[56] Wilson,G. C. ,& Sally,B. (2001). Developmentally appropriate Logo computer programming with young children[J]. Information Technology in Childhood Education,8(1):229-245.

[57] Zelazo, P. , Carter, A. S. , Reznick, J. S. , & Frye, O. (1997). Early development of executive function: A problemsolving framework[J]. Review of General Psychology,1(2):198-226.

(编辑:魏志慧)

Research Status and Action Path on International Children Programming Education

SUN Lihui & ZHOU Danhua

(*School of Education*, *Tianjin University*, *Tianjin* 300072, *China*)

Abstract: *Since* 1968, *there have been many programming tools including Logo*, *Tangible Programming*, *Scratch in the field of children programming education. From the perspective of programming form*, *Logo programming language is more like the text-based real programming language*, *which has a more powerful enlightening effect on computer programming*; *Tangible Programming gets rid of the single programming environment by computers*, *involves children in programming activities in real physical space and helps younger children understand and master programming knowledge*; *Scratch programming environment makes programming more visual*, *which could make children design and arrange the behavior of target roles by dragging and dropping programs blocks with different functions by mouse. Although it is somewhat different from real programming activities*, *its operating mechanism and design concept are consistent with the abilities trained by programming activities. Moreover*, *Scratch is easy to handle*, *which can enhance children's enjoyable experience in programming learning and allows children to create their own projects easily. Based on the classification and summary findings from the perspective of theoretical ability improvement*, *Logo programming language is of great help to the improvement of children's cognitive ability*; *Tangible Programming focuses on improving children's sequencing ability*; *Scratch is related to the improvement of computational thinking ability. The international children programming education practice applications mainly from three aspects of the students*, *the teachers*, *the activities have carried on massive experimental researches. As for student dimension*, *students' learning interest*, *motivation*, *and self-efficacy are mainly concerned. The teacher dimension focuses on teachers' classroom role*, *teaching experience*, *teaching style and comfort level of technology use. The activity form dimension advocates diversified activities*, *such as collaborative learning and parent-child cooperation. Based on this*, *the implementation of children's programming education in China should be based on a correct understanding of the true meaning of children's programming education. Schools*, *especially public schools*, *should carry out extensive experimental research. In addition*, *the promotion of children's programming education needs strong support from government. It is particularly important to select and develop children's programming tools which are suitable for children's characteristics in China. Children's programming textbooks should be compiled as soon as possible in combination with subject content. And a teaching team should be built systematically with programming thinking skills.*

Key words: *children programming education*; *practical application*; *action path*