A University of Sussex PhD thesis

Available online via Sussex Research Online:

http://sro.sussex.ac.uk/

This thesis is protected by copyright which belongs to the author.

This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the Author

The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the Author

When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given

Please visit Sussex Research Online for more information and further details
EMBODIED APPROACHES TO LEARNING PROGRAMMING

BY ABRAR ALMJALLY

Supervised by Kate Howland

A thesis submitted in fulfilment of the requirements of the degree of
Doctor of Philosophy
In the
School of Engineering & Informatics

Jun 2022
DECLARATION

I hereby declare that this thesis is my own work and carried out in accordance with the requirements of the University’s Regulations and Code of Practice for Research Degree Programmers. It has not been submitted for any other academic award.

SIGNED: ___________________________ DATE: ___ / ___ / _____
SUMMARY

This thesis explores observable representations of embodied learning, such as physicality, gestures and the use of conceptual metaphors among students in primary computing education. To understand the influence of more physically interactive forms of interface, I compared the use of two user interfaces – a Tangible User Interface (TUI) and Graphical User Interface (GUI) – to foster programming skills in primary school students aged between six and seven. The first phase of this thesis, Studies 1 and 2, adopted a between-subjects design to examine the impact of interface type on several variables including enjoyment, attitudes, learning outcomes, and frequency of gestures, as well as the effect of gender on each variable. Both Study 1 and 2 examined the relationship between student gestures and learning outcomes. Study 1 examined the effect of physicality as an input by asking students to use a block-based programming environment to control a physical robot (PR) to solve six activities (two complex, four simple) where the students used either a TUI-PR or a GUI-PR. The use of a GUI-PR was associated with improved learning outcomes, but the TUI-PR led to a greater attitudinal improvement toward computing. No difference was identified in the number of gestures used by participants in the TUI-PR and GUI-PR groups, but a statistically significant difference was identified between the mean learning gains in programming of high-frequency gesturers and low-frequency gesturers, with the top quartile showing significantly greater learning gains.

Study 2 further examined the effect of physicality as both an input and an output by comparing two block-based programming environments: first, a TUI-PR consisting of physical, hand-manipulated blocks to control a physical robot; and second, a GUI-SR, which involve using touchscreen-operable programming blocks to control an on-screen robot (SR) to solve four simple activities. No difference was observed between the TUI-PR and GUI-PR in terms of learning outcomes, but the GUI-PR was associated with
attitudinal improvement toward computing. Additionally, no difference was observed in the number of gestures used by participants in the TUI-PR and GUI-SR groups and no relationship was identified between the frequency of gestures and learning gains. In both studies, no difference was found in terms of the level of enjoyment or by gender across all the measures. The results also demonstrated that children used a range of gestures to represent the concept of iteration including pointing, literal representational gestures and metaphorical representational gestures.

In Study 3, we addressed a gap in the current theoretical understanding of computing education by drawing on embodied cognition theory. Using methodological tools from cognitive linguistics and gesture research, an analysis of how primary school students used spontaneous co-speech gestures when responding to interview questions and describing programming concepts was conducted. The findings show representational patterns in these gestures, thereby suggesting the potential of this methodological approach to provide a deeper understanding of the nature of learners’ cognition in the domain of computing education.

This work that contributes to two main areas: first, the field of interaction design, particularly relating to the importance of physicality in programming environments for children; and second, current understandings of the importance of gestures and conceptual metaphor in CS education in primary school.

This thesis presents an in-depth comparison of the use of a TUI and GUI to teach programming skills to primary school students. In particular, the findings indicate that a GUI-SR is suitable for children’s learning and is associated with greater attitudinal improvement than a TUI. This thesis also investigated the potential relationship between increased embodiment in the interface and output device (e.g., physicality) and increased use of embodied representations (e.g., gestures) that showed no relationship across Studies 1 and 2.

This research describes children’s use of spontaneous gestures when solving programming problems and explaining programming concepts. Additionally, regarding
the use of spontaneous gestures, this research demonstrates how investigating children’s gestures may help to characterise children’s conceptions in primary computing, possibly allowing the identification of misconceptions and assisting the identification of productive educational strategies. This research has also provided evidence indicating that children use spontaneous hand gestures to demonstrate abstract computational concepts, even in the absence of relevant stimuli (i.e., written code); this reflects how gestures may indicate the embodiment of the children’s computing notions. Furthermore, this research presents tentative evidence of cultural influences on embodied conceptualisations. The findings suggest that the direction of a culture’s written language (i.e., right-to-left or left-to-right) influences the direction and use of conceptual metaphors in CS. Finally, this research identified a positive relationship between the mean learning gains of high-frequency gesturers and low-frequency gesturers on tasks with varying problem difficulties.

This work represents the first step toward understanding children’s embodied descriptions of programming and the potential role of gestures in supporting their learning. This was a worthwhile area of research because, although the analysis of children’s gestures has already proven valuable in illuminating knowledge acquisition and conceptual development in science, technology, engineering, and mathematics (STEM) fields such as mathematics itself, the area of computing education is currently underexplored.
ACKNOWLEDGEMENTS

I would like to thank Allah for supporting me in completing my dissertation. Successfully finishing this task would not have been possible without all of His blessings.

I have been lucky to have had two amazing supervisors: Dr Kate Howland who never hesitated to give valuable advice, guidance, and encouragement, providing support for my journey from day one and offering infinite patience.

I would also like to thank my second supervisor, Prof. Judith Good, who taught me how to consider the big picture in my research; her experiences illuminated my way of thinking. Additionally, my gratefulness also extends to Prof. Ben du Boulay for his guidance and feedback. He acted as my additional supervisor whenever I needed further discussion or support.

I offer my deep gratitude to Kate, Judith, and Ben. You all taught me how to work in a team and function as an independent researcher, which will continue to benefit me for many years to come.

I am also thankful for the efforts of my thesis committee members, Dr Charlotte Robinson, and Dr Andrew Manches. Both provided their incredibly valuable time to examine my dissertation and, have made suggestions that have helped enhance the presentation of my thesis.

I am appreciative of my classmates at the HCI lab for their support and valuable feedback, especially my officemates James Jackson, Norah Sarhan, Grazia Ragone, and Anthony Trory. We had many great moments, with our brief chats regularly keeping me motivated throughout the four years. I will
often look back happily on our afternoon lab meetings, coffee chats, and lunches.

Immeasurable thanks are due to my parents. Together, you two have always reinforced my confidence and helped sustain my ambition. To my Mum ‘Sheikha Altowaim’, thank you for coming with me to Brighton in my first years and helping take care of Raseel, my daughter. I am thankful for your love and continuous prayers for me! To my dad ‘Abdulaziz Almjally’, thank you for always insisting on helping and supporting me even when you were at your busiest! Your phone calls and endless concern have been my motivation to make this journey!

I would like to thank my amazing brother, Fasisal Almjally, for being my roommate and companion on this journey. Fasisal can make me laugh even on the toughest days, and I could not imagine being in the UK without him!

Thank you to my only sister, Mjd, for her visits to the UK. I am very appreciative of all the efforts she has made to support me as much as she can.

Also, thanks to my uncle Abdullah Almjally for his endless support, he was present during my master’s and PhD journeys, thank you for believing in me.

Most especially, I thank my husband, Abdullah Alghonim, for supporting me every step of the way. Abdullah, you have inspired me to reach my full potential.
ABBREVIATIONS AND KEY TERMS

Computer science (CS): The study of computers and the theoretical and practical applications of computing, including programming.

Embodied cognition (EC): A set of theories based on the idea that human cognitive and linguistic processes are grounded in the human body’s perceptual and physical interactions with the environment.

Child-computer interaction (CCI): The scientific field that focuses on studying the interaction between children and technology.

Science, technology, engineering, and mathematics (STEM): it is a curriculum of learning combined Science, technology, engineering and mathematics.

Physically: Having physical materials present and addressing the effect of manipulating a tangible representation, and having a physically engaging output, such as a robot.

Tangible user interface (TUI): A user interface in which a user interacts with digital information (e.g., a programmable robot) through the physical environment (e.g., tangible programming blocks).

Graphical user interface (GUI): A user interface in which a user interacts with visual components for a computer application through a screen.

Action: A movement of the body, it can be to accomplish something or performed on a physical object or making a hand movement such as gesture.

Gesture: The hand movements that people produce when talking in which they porty action or object (iconic/literal gestures), represented an abstract idea (metaphorical gesture), or to referral to location or item (pointing/deictic gesture). Gesture are kind of action in which they involve movements of the body.

Abstract concepts: Concepts that do not have physical constraints as they have no concrete representation in the physical world (e.g., emotions, metaphors, and thinking as an abstract action).

Concrete concept: Concept that refers to entities with a presentation in the physical world and special constraints, such as the Sun in the sky or walking is a concrete action.

Programming: The process of producing a set of instructions that tells a computer how to complete a task.

Iteration: A process used in programming wherein a step, instruction, or command is repeated. In the programming environment studied in this thesis, iteration is supported through a repeat block that repeats a piece of code twice.
# Table of Contents

Chapter 1 ................................................................................................................................. 1

1.1 Introduction ......................................................................................................................... 1

1.2 Research Questions ............................................................................................................. 3

1.3 Motivation for Research ...................................................................................................... 4

1.3.1 Gender ............................................................................................................................ 7

1.4 Embodiment Approach ....................................................................................................... 7

1.4.1 Tangible User Interfaces (TUI) ....................................................................................... 9

1.4.2 Gestures and Conceptual Metaphor ................................................................................. 11

1.5 Research Contributions .................................................................................................... 13

1.6 Thesis Structure ................................................................................................................. 14

1.7 Related Publications ......................................................................................................... 15

1.8 Summary of Research Activities ....................................................................................... 16

Chapter 2 ................................................................................................................................ 17

2.1 Introduction ......................................................................................................................... 17

2.2 Computing in Primary Schools in Saudi Arabia ................................................................. 17

2.3 Computing in Primary Schools in the UK ......................................................................... 18

2.3.1 Curriculum in the UK .................................................................................................. 19

2.3.2 Programming Concept for Age 6 to 7 ....................................................................... 20

2.3.3 Student Enrolment in Post-Compulsory CS ................................................................. 22

2.3.4 Primary School Teachers’ Perspectives ...................................................................... 22

2.3.5 Programming Environments in Primary Schools ......................................................... 23

2.4 Embodied Cognition .......................................................................................................... 26

2.4.1 The Cognitive Linguistics Perspective on Embodied Cognition ................................. 26

2.4.2 The Cognitive Science Perspective on Embodied Cognition ..................................... 27

2.4.3 The Human-computer Interaction (HCI) Perspective on Embodiment ................... 28

2.5 Educational Approaches Based on Embodied Cognition ................................................. 28

2.5.1 Physical Manipulation ................................................................................................. 29

2.5.2 Image Schemas and Action ........................................................................................ 30
2.5.3 Conceptual Metaphors ................................................................. 31
2.5.4 Gestures .................................................................................... 32
2.6 Roles of Gestures in Learning ......................................................... 33
  2.6.1 Enhancing Problem-solving and Reducing Cognitive Load .......... 33
  2.6.2 Generating New Ideas ............................................................... 34
  2.6.3 Revealing Implicit Knowledge .................................................. 35
  2.6.4 Effect of Teacher Gesturing on Learning .................................... 36
2.7 Embodiment and Computing .......................................................... 38
  2.7.1 Embodiment and Learning Programming .................................... 40
  2.7.2 Blocks Domain Analysis ............................................................ 41
2.8 Empirical Work Investigating the Effect of Physicality (TUI and GUI) and Gestures .......................................................... 42
  2.8.1 Comparing TUIs and GUIs for Children's Programming ............. 42
  2.8.2 Exploring the Role of Gestures in Learning Computer Science ...... 47
2.9 Conclusion ..................................................................................... 48

Chapter 3 ........................................................................................ 50
3.1 Introduction .................................................................................... 50
3.2 Methods ......................................................................................... 51
  3.2.1 Participants and Setting ............................................................ 52
  3.2.2 Materials .................................................................................. 53
  3.2.3 Learning Activities ................................................................. 63
  3.2.4 Procedure ................................................................................ 65
3.3 Analysis ......................................................................................... 67
  3.3.1 Test Schema ............................................................................ 67
  3.3.2 Overview of Gesture Coding Rubrics ......................................... 67
3.4 Results .......................................................................................... 70
  3.4.1 Effect of Interface Type and Gender on Learning Outcomes, Attitudes and Enjoyment .......................................................... 70
  3.4.2 Overview of Time Spent on Session, Interaction and Attempts ....... 73
  3.4.3 Use of Spontaneous Gestures While Solving Programming Tasks.... 75
3.5 Discussion ................................................................. 81
  3.5.1 RQ1 What are the Key Differences in How TUI and GUI Programming Environments Support the Development of Programming Skills for Students Aged 6-7 Years Old? ................................................................. 81
  3.5.2 RQ2 How Do Young Children Use Spontaneous Gestures While Learning Programming and What is the Relationship Between Gesturing and Learning Outcomes? ........................................ 83
3.6 Limitations ..................................................................... 86
3.7 Conclusion ..................................................................... 87
Chapter 4 ............................................................................ 88
  4.1 Introduction ................................................................... 88
  4.2 Methods ......................................................................... 89
    4.2.1 Participants and Setting ........................................... 90
    4.2.2 Materials ............................................................... 91
    4.2.3 Procedure ............................................................. 94
  4.3 Analysis .......................................................................... 95
    4.3.1 Test Schemas ......................................................... 95
    4.3.2 Overview of Gesture Coding Rubrics ......................... 95
  4.4 Results .......................................................................... 95
    4.4.1 Effect of Output Physicality and Gender on Learning Outcomes, Attitudes and Enjoyment ......................................................... 96
    4.4.2 Overview of Learning Session Data ........................... 98
    4.4.3 Use of Spontaneous Gestures While Solving Programming Tasks ...... 101
  4.5 Discussion ..................................................................... 107
    4.5.1 RQ1 What are the Key Differences in How TUI And GUI Programming Environments Support the Development of Programming Skills for Students Aged 6-7 Years Old? ................................................................. 107
    4.5.2 RQ2 How Do Young Children Use Spontaneous Gestures While Learning Programming and What is the Relationship Between Gesturing and Learning Outcomes? ........................................ 108
  4.6 Conclusion ..................................................................... 113
Chapter 5 ............................................................................ 115
5.1 Introduction ........................................................................................................ 115
5.2 Method .............................................................................................................. 116
  5.2.1 Participants ................................................................................................. 117
  5.2.2 Data Collection .......................................................................................... 117
  5.2.3 Immediate and Delayed Interviews ......................................................... 118
  5.2.4 Interview Questions .................................................................................. 119
  5.2.5 Procedure and Materials .......................................................................... 120
  5.2.6 Measures and Data Analysis ...................................................................... 121
5.3 Results ............................................................................................................... 123
  5.3.1 Overview of the Questions’ Responses ..................................................... 123
  5.3.2 Types of Spontaneous Gesture when Explaining Programming Concepts 126
  5.3.3 Types of Conceptual Metaphor Used to Explain Programming Concepts 127
  5.3.4 Representational Gesture Changes Over Time ....................................... 134
  5.3.5 The Effect of The Interface Type on The Use of Gesture When Explaining Programming Concepts ................................................................. 144
5.4 Discussion ......................................................................................................... 144
  5.4.1 RQ3.1 What Types of Spontaneous Gesture Do Young Children Use to Explain Programming Concepts? ................................................................. 145
  5.4.2 RQ3.2 What Types of Conceptual Metaphor Do Young Children Use to Explain Programming Concepts? ................................................................. 146
  5.4.3 RQ3.3 Are There Any Differences Evident in The Use Of Gestures and Conceptual Metaphors When Children Are Interviewed Again Two Weeks After The Introductory Programming Activity Vs. The First Interview? ........................................................................ 149
  5.4.4 RQ3.4 What is The Effect of Interface Type (GUI Versus TUI) on The Use Of Spontaneous Gestures When Explaining Programming Concepts? .......... 150
5.5 Implications ..................................................................................................... 150
5.6 Conclusion ....................................................................................................... 151

Chapter 6 .............................................................................................................. 152
  6.1 Conclusions .................................................................................................. 152
  6.2 Contributions ............................................................................................... 158
    6.2.1 Interaction Design-related Contributions ............................................. 159
    6.2.2 Gesture and Conceptual Metaphor-related Contributions ..................... 160
6.3 Limitations ................................................................................................................. 162

6.4 Implications and Future Work .................................................................................. 164

   6.4.1 Interfaces for Learning Programming ................................................................. 164

   6.4.2 The Role of Gestures in Learning, and Problem-Solving ................................. 165

   6.4.3 The Role of Gestures in Communication and Explaining ................................. 167

   6.4.4 The Role of Metaphors in Communication, Learning and Explaining .......... 167

   6.4.5 Teachers and Gestures ......................................................................................... 168

References ....................................................................................................................... 170

Appendix A ....................................................................................................................... 191

Appendix B ....................................................................................................................... 199

Appendix C ....................................................................................................................... 201
# Table of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>TUI setting (left) and GUI (right)</td>
<td>54</td>
</tr>
<tr>
<td>3.2</td>
<td>mBot with orange arrow and activity sheet (left) and tile types (right)</td>
<td>55</td>
</tr>
<tr>
<td>3.3</td>
<td>mBot robot and robot mat</td>
<td>56</td>
</tr>
<tr>
<td>3.4</td>
<td>Question 1</td>
<td>58</td>
</tr>
<tr>
<td>3.5</td>
<td>Question 2</td>
<td>59</td>
</tr>
<tr>
<td>3.6</td>
<td>Question 3</td>
<td>60</td>
</tr>
<tr>
<td>3.7</td>
<td>Question 4</td>
<td>60</td>
</tr>
<tr>
<td>3.8</td>
<td>Question 5</td>
<td>61</td>
</tr>
<tr>
<td>3.9</td>
<td>Stickers used in attainment</td>
<td>62</td>
</tr>
<tr>
<td>3.10</td>
<td>Learning activities</td>
<td>65</td>
</tr>
<tr>
<td>3.11</td>
<td>Summary of results by condition (a) Learning gain; (b) Attitudinal change;</td>
<td>71</td>
</tr>
<tr>
<td></td>
<td>(c) Enjoyment score</td>
<td></td>
</tr>
<tr>
<td>3.12</td>
<td>P1</td>
<td>79</td>
</tr>
<tr>
<td>3.13</td>
<td>P2</td>
<td>79</td>
</tr>
<tr>
<td>3.14</td>
<td>P3</td>
<td>80</td>
</tr>
<tr>
<td>3.15</td>
<td>P4</td>
<td>80</td>
</tr>
<tr>
<td>3.16</td>
<td>P4</td>
<td>80</td>
</tr>
<tr>
<td>3.17</td>
<td>P5</td>
<td>81</td>
</tr>
<tr>
<td>3.18</td>
<td>P6</td>
<td>81</td>
</tr>
<tr>
<td>4.1</td>
<td>Study location</td>
<td>91</td>
</tr>
<tr>
<td>4.2</td>
<td>TUI setting (left) and mBot robot and robot mat (right)</td>
<td>92</td>
</tr>
<tr>
<td>4.3</td>
<td>GUI programming blocks (left) and on-screen robot (right)</td>
<td>92</td>
</tr>
<tr>
<td>4.4</td>
<td>System tiles</td>
<td>93</td>
</tr>
<tr>
<td>4.5</td>
<td>Learning activities</td>
<td>94</td>
</tr>
<tr>
<td>4.6</td>
<td>Summary of results by condition (a) Learning gain; (b) Attitudinal change;</td>
<td>98</td>
</tr>
<tr>
<td></td>
<td>Enjoyment score</td>
<td></td>
</tr>
<tr>
<td>4.7</td>
<td>P01</td>
<td>106</td>
</tr>
<tr>
<td>4.8</td>
<td>P05 action inside the repeat (right)</td>
<td>107</td>
</tr>
<tr>
<td>5.1</td>
<td>(a) Q3 - Simple code aimed to describing how a simple program works; (b)</td>
<td>121</td>
</tr>
<tr>
<td></td>
<td>Repeat block; (c) Q5 - Complex code aimed to describing how a complex program works</td>
<td></td>
</tr>
</tbody>
</table>
Figure 5.2 Gesture type illustrations (a) Participant pointing to the code; (b) Participant using RG-literally to describe the left turn; (c) Participant moving their hand in a circular motion to represent repeat.

Figure 5.3 Motion gesture examples (a) Pinch; (b) Container (c) Longitudinal movement; (d) Frontal movement.

Figure 5.4 Transversal axis examples (a) One big circle, clockwise; (b) Drawing two circles clockwise on the table; (c) Moving both hands at the same time in an anticlockwise motion, three times.

Figure 5.5 Transversal axis examples (a) Grasping with a small circle (clockwise); (b) Arc (clockwise); (c) Two big circles (anticlockwise).

Figure 5.6 Transversal axis examples (a) Waves (anticlockwise); (b) Right-to-left movement (anticlockwise); (c) Two circles (clockwise and anticlockwise).

Figure 5.7 P45 use of gesture to reveal implicit knowledge.
TABLE OF TABLES

Table 1.1 Summary of research achievement .......................................................... 16
Table 2.1 Studies evaluating TPLs for children ...................................................... 42
Table 3.1 Group characteristics ............................................................................. 53
Table 3.2 Group characteristics for available videos .............................................. 67
Table 3.3 Gesture type ............................................................................................ 68
Table 3.4 Gesture location ....................................................................................... 69
Table 3.5 Gesture purpose ....................................................................................... 69
Table 3.6 Descriptive analysis of times and attempts by condition (N is the number of pairs) ........................................................................................................... 73
Table 3.7 Descriptive analysis of times and attempts by condition and gender (N is the number of pairs) ......................................................................................... 74
Table 3.8 Researcher and participants gesture types ................................................ 77
Table 3.9 Researcher and participants gesture’s locations ......................................... 77
Table 3.10 Cross-tabulation for gestures type and purpose ..................................... 78
Table 3.11 Iteration representational gesture description ........................................ 79
Table 4.1 Group characteristics ............................................................................. 91
Table 4.2 Group characteristics for available videos .............................................. 95
Table 4.3 Overall descriptive session data for the two condition groups (N is the number of pairs) ................................................................................................. 99
Table 4.4 Overall descriptive session data for the two condition groups divided by gender (N is the number of pairs) ........................................................................ 99
Table 4.5 Gesture’s statistic across activities in Study 1 .......................................... 104
Table 4.6 Gesture’s statistic across activities in Study 2 .......................................... 104
Table 4.7 Cross-tabulation for gesture type and purpose ..................................... 105
Table 4.8 Iteration representational gesture description ......................................... 106
Table 5.1 Interview questions ............................................................................... 119
Table 5.2 Prompts and responses for each question ............................................. 124
Table 5.3 Summary of noise representational gesture .......................................... 136
Table 5.4 Representational- metaphorical gesture for repeat ................................. 138
CHAPTER 1
INTRODUCTION AND MOTIVATION

1.1 INTRODUCTION

This thesis is motivated by the belief that exploring the role of Embodied Cognition (EC) in computing education will lead to a greater understanding of children's development of programming concepts and benefit computing education, for which there is increased demand worldwide (Heintz, Mannila, and Farnqvist 2016; Ministry of Education; Tatweer Company for Educational Services 2018; Sentance and Csizmadia 2017). Concerns have been raised regarding the theoretical evidence base for computing education (Grover and Pea 2013), highlighting the need to develop our theoretical and practical understanding of how best to teach computing concepts to younger children. In domains such as language learning, physics and mathematics, learning environment design and implementation have been informed by theories of EC. These emphasise the importance of the role of physicality and action, such as object manipulation and gestures (Fugate, Macrine, and Cipriano 2019). Many teachers believe in the effectiveness of physicality in the computing classroom (Sentance and Csizmadia 2017), although the evidence is mixed (Horn, Crouser, and Bers 2012). Very few studies have investigated the potential of gestures in learning computing or in reflecting children's understanding (Manches et al. 2020).

The perspective of EC is underexplored in the literature on child-computer interaction (CCI) despite the gradual acceptance of theories of EC in the field of cognitive science. However, EC offers a valuable lens into computing education, including in children’s early years, because when designers develop screens, user interfaces and other tangible devices, EC can provide insights into how more actions and physically interactive forms of interface can support cognitive development and children’s learning and play. An EC perspective can help to examine the differences in physicality and actions in type, amount, directness, and conceptual mapping of different interfaces (TUI and GUI). Indeed, as
evolving technologies blur the difference between physical and graphical interfaces, it is important to develop the understanding of how different forms of interaction and action affect children’s conceptual development (Manches and Price 2011).

In the domains of both psychology and human-computer interaction (HCI), researchers (e.g., Healy and Antle, respectively) have produced evidence for the significance of action and the environment in influencing children’s cognitive abilities. In the case of Healy, the author emphasised the role of physicality in children’s lives. In particular, overuse of screens can limit children’s engagement in the types of kinaesthetic activities that can advance their awareness of relationships in the world, having an impact on their cognitive development (Healy 1998). Antle has provided evidence that tangible manipulation can simplify cognitive computation, enhance understanding and improve memory recall (Antle 2013). However, the mechanisms behind which actions support learning are often unclear.

This thesis examines the role of embodiment in computing education represented in physicality that allow direct interaction with representation (tangible programming blocks) and having a physical output (physical robot) through tangible user interfaces (TUIs) and action including gestures and conceptual metaphor. First, the thesis investigates whether TUIs improve learning outcomes, enjoyment and attitudes toward computing in comparison with graphical user interfaces (GUIs) and whether gender differences affect these measures. The gender aspect is addressed because the classes in Saudi Arabia (where the empirical studies were conducted) are single-gender, which may accentuate differences. Also, women globally are underrepresented in the field of STEM, including computing (Kayan-Fadlelmula et al. 2022). It is important to diversify technology related occupations among gender to ensure that creating new technologies includes all the groups that intended to serve and to increase the qualified labour of women. Following this, the use of gestures in learning activities in computing is investigated. The thesis then investigates how young children use conceptual metaphors, as externalised through gestures, to explain computing concepts.
1.2 Research Questions

This doctorate research explores the role of embodiment in computing education for younger children (aged six-to-seven) in primary schools in Saudi Arabia. The thesis investigates the use of physicality by comparing a TUI and a GUI for learning programming skills. The thesis also examines broader factors beyond learning outcomes that are relevant in computing education, such as attitudes towards computing and enjoyment. These factors are particularly important at the primary school level because they may affect children's later decisions to choose a career in computer science (CS).

As part of investigating the theoretical evidence base for computing education, the thesis examines the role of interface type in children's gestures and the role of gestures in learning. Additionally, it explores the conceptual metaphors which children use to describe computing concepts by analysing their spontaneous gestures. The thesis adopts an EC lens to explore the development of programming concepts, which is a novel approach in CCI.

Specifically, the thesis addresses the following questions:

**Research Question 1 (RQ1):** What are the key differences in how TUI and GUI programming environments support the development of programming skills for students aged six-to-seven?

- **RQ1.1:** Are there any differences in learning outcomes, attitudes toward computing and enjoyment of computing?
- **RQ1.2:** Are there any differential gender effects in the above measures?

**Research Question 2 (RQ2):** How do young children use spontaneous gestures while learning programming and what is the relationship between gesturing and learning outcomes?

- **RQ2.1:** How does interface type affect children’s use of spontaneous gestures while completing programming tasks?
• **RQ2.2:** How are young children's learning outcomes related to the frequency of spontaneous gestures while completing programming tasks?

• **RQ2.3:** What are the types and purposes of young children's spontaneous gestures while completing programming tasks?

• **RQ2.4:** How are gestures used to represent an abstract programming concept (iteration)?

**Research Question 3 (RQ3):** How are conceptual metaphors, as externalised through gestures, used by young children to explain programming concepts?

• **RQ3.1:** What types of spontaneous gestures do young children use to explain programming concepts after an introductory programming activity?

• **RQ3.2:** What types of conceptual metaphor do young children use to explain programming concepts?

• **RQ3.3:** Are there any differences evident in the use of gestures and conceptual metaphors when children are interviewed again, two weeks after the introductory programming activity, compared with that in a first interview immediately after the activity?

• **RQ3.4:** What is the effect of interface type (GUI versus TUI) on the use of spontaneous gestures when explaining programming concepts?

### 1.3 Motivation for Research

In children's computing education, there is global interest in extending computing education at the primary level. There are also key questions about the effectiveness of physicality (Sentance and Csizmadia 2017), action including gestures and the lack of a theoretical evidence base for computing education (Grover and Pea 2013). In addition, there are questions around the effect of different interfaces across genders, given that females are underrepresented in computing. These issues form the primary motivation for this research. The thesis approaches these issues by investigating the role of embodiment in computing education represented in physicality (TUIs), gestures and conceptual metaphor.
In 2006, Wing prompted a debate on the importance of computational thinking for a wider audience beyond CS (Wing 2006). Since then, computing concepts including programming have been more widely taught and several countries have introduced CS into their school curricula. Saudi Arabia has yet to start teaching computing to primary school students, therefore children at age six-to-seven years old have not yet been exposed to programming activities. This, combined with the researcher’s knowledge of the Saudi Arabia school system and teaching practices, made the context a suitable one in which to explore the role of EC in the development of programming concepts. Much of the literature in this thesis is on research carried out in the UK because it is one of the first countries to adopt a compulsory nationwide computing curriculum for primary schools and is the country in which the doctoral research was supervised.

Several studies have found that children can learn programming from early childhood (Bers 2010; Fessakis, Gouli, and Mavroudi 2013). Programming can contribute positively to cognitive development, especially between the ages of three-to-six (Manches and Plowman 2017). Early elementary grade is a suitable time to introduce basic STEM (Science, technology, engineering, and mathematics) skills for young children, as children at that age are most eager to learn (Nadelson et al. 2013). In Saudi Arabia first grade students are aged 6 to 7 years old therefore, this research investigated this age group.

In 2017, Robertson et al. carried out a study investigating children's knowledge of computers with a total of 18 children between 5 and 8 years old from two Scottish primary schools. The children were interviewed about their understanding of what computers are, how they are programmed and their beliefs about them. They provided knowledgeable answers about computer usage, but they could not explain how computers work and they had varied theories about whether or not computers could think. The authors proposed fundamental concepts that young children could learn to enhance their knowledge of computers, as children need to know the limits and capabilities of technology. Current knowledge of computing functionality might benefit children's everyday lives. It is thus important for children to know both the advantages and
disadvantages of being technology-dependent and relying on computers (Robertson, Manches, and Pain 2017).

In addition, children need to use technology that is suitable for their cognitive development, allowing them to engage with their environment. Integrating computing with their school or everyday teaching materials and curricula, for example, is vital. A study by Bers et al. in 2014 explored the possibility of engaging kindergarten students (n=53) in learning programming using tangible interfaces. These researchers developed the *TangibleK Robotics* curriculum (Bers 2010), which contains 20 hours of instruction and a final project to assist children in understanding programming concepts. The study revealed that when provided with age-appropriate technologies, curricula and education, young children eagerly participate in learning computer programming and take the early steps to develop computational thinking. However, Sipitakiat et al. (2013) found that children younger than eight years old find it difficult to understand basic robotics concepts. For example, they struggled to choose the correct blocks to turn the robot left or right, to understand the association of the symbols and the needed action and to adjust the numbers in the blocks to increase or decrease the amount of turn (Sipitakiat and Nusen 2012). Therefore, further research is needed to explore how young children use different programming tools and their effects on problem-solving and computational thinking abilities (Sipitakiat and Nusen 2012).

Computing is now a compulsory subject in primary and secondary education in the UK. The primary school *Tech Literacy Report 2016*, which reports on interviews with 400 primary school teachers in England, Scotland and Wales and students from five schools (Craft et al. 2014), showed that teachers think that technical literacy is vital for their students. 78% believe that it is as important as reading and writing and 96% agree that it is in high demand in the job market. Regarding their role in preparing students for the digital world, 25% of teachers strongly agree that they can prepare children for the digital world and 57% understand computational thinking, but 35% find computing difficult to teach (BT and Ipsos MORI 2016). Teachers struggle with teaching computing in terms of their own computing subject knowledge (such as programming) as well as in problem-
solving; namely their educational approach to supporting students in thinking through a problem, trying to break it down and to trace errors (Sentance and Csizmadia 2017). Some teaching support for the UK computing curricula is available online in the form of workshops and materials, such as Barefoot Computing (https://barefootcas.org.uk/) and Quickstart Computing (http://quickstartcomputing.org), but most teachers do not have degrees in computing.

To ensure all students can understand and apply the fundamental principles and concepts of CS, more empirical work is needed to develop knowledge in interaction design concerning the importance of the interfaces used in CS education for children aged six-to-seven years.

1.3.1 Gender

Girls and boys may become equally involved in this computational learning activity if physical programming is introduced as an activity conducted apart from a typical computer. Making programs available to younger children while gender differences are less developed has potential to help to close the "gender gap" in later grades (KIPR 2022).

Additionally, the literature on women in STEM indicates that women frequently avoid computing careers. For instance, the number of females interested in a career in computers decreases at each step of the educational system, starting from primary school and through to higher education and the PhD level (Camp 1997; Hope 2021). To improve attitudes toward computing, it is important to measure these attitudes in children while programming; this can provide rich information about the benefits, opportunities, and risks associated with different programming environments (Imhof, Vollmeyer, and Beierlein 2007).

1.4 Embodiment Approach

In the above section, it was noted that teachers report needing more support in the classroom to understand children's abilities and to develop computing concepts, which will eventually enhance young children's learning.
The literature provides evidence of the possibility of using embodiment in the form of TUIs, gestures and conceptual metaphors to improve pedagogy (JUNG 1991) by helping learners to connect sensory representation with abstract concepts, which is vital in learning (Weisberg and Newcombe 2017). Abstract concepts refers to concepts that do not have physical constraints as they have no concrete representation in the physical world (e.g., emotions, metaphors, and thinking as an abstract action). (K. Dijkstra et al. 2014). In contrast, concrete concepts refer to entities with a presentation in the physical world and special constraints, such as the Sun in the sky or walking as a concrete action.

There is evidence that actions have an important role in improving the development of children's thinking skills. For example, understanding abstract symbols used in everyday education, such as words in reading comprehension (Glenberg et al. 2004; Glenberg, Goldberg, and Zhu 2010) and numbers in maths (Goldin-Meadow, Cook, and Mitchell 2009), which can be developed via embodied interaction to foster learning. There has been little empirical work conducted to explore why and how different interfaces and activities may enhance the learning of programming (Sapounidis and Demetriadis 2017), particularly regarding the role of using embodied interaction such as object manipulation and spontaneous gesturing. Physicality using tangible manipulation might trigger affordances for action that facilitate retrieval from memory (Antle and Wise 2013).

Embodied tools have the potential to enhance learning in science, technology, education and mathematics (STEM) disciplines. Students can gain from incorporating embodied tools because STEM disciplines depend on representation systems that require sensory encoding (such as data visualisation) and high abstraction (such as mathematical formulae or programming code). Therefore, learners need to understand how to link sensory representation with abstraction. Second, embodied learning tools can enhance students' abilities to connect the abstract to the concrete and enhance their memory and cognitive skills, such as strategic or spatial cognition; the reasoning abilities used to solve problems (Clifton et al. 2016; DeSutter and Stieff 2017).
Current efforts in computing education need additional input from theoretical and empirical research (Weisberg and Newcombe 2017). Educational research shows that motor activity can support the retention of learned concepts because it supports extra cues that represent and recall knowledge (Carbonneau, Marley, and Selig 2013; Chu and Kita 2011; Lee 2014). Instead of only visualising or hearing information, supporting information interactions with action or gesturing can cause deeper levels of processing that create stronger memory traces, allowing learners to activate multiple avenues for retrieving the memory later (Craik and Lockhart 1972).

In computing education, most of the studies that address EC approaches (such as object manipulation using TUIs to enhance computing learning) do not address the particular role of physicality in learning, nor how it could be beneficial. Moreover, few studies provide empirical evidence of the learning outcomes of programming activities. Furthermore, there is little research addressing the role of spontaneous gestures in computer education and few studies have examined the relationship of spontaneous gestures to students' learning.

1.4.1 Tangible User Interfaces (TUI)

Using physicality, especially for young children, has motivated the development of TUIs which have great potential to support learning. There are many potential benefits of adopting TUIs. For example, they can offer a natural and immediate form of interaction that is accessible to learners (Marshall 2007). They can promote active and hands-on engagement, exploration, discovery and reflection (Marshall, Price, and Rogers 2003) and give learners ‘tools to think with’ (Resnick 1998) that enable them to learn abstract concepts through concrete representations, plus supporting collaborative activity among learners (Antle 2007). TUIs, compared with other interfaces, might be the most suitable for engaging children (Kahn 1996). Piagetian developmental theory emphasises the value of the manipulation of concrete physical objects in supporting and developing thinking, in particular for young children. Because TUIs often utilise concrete physical manipulation, they might support more effective natural learning (Sluis et al. 2004; Terrenghi et al. 2006; Zuckerman, Arida, and Resnick 2005). TUIs have facilitated
learning about diverse topics and domains such as ratios and computing hardware (Abrahamson and Sánchez-García 2016; Crease 2006).

This doctoral research examines the role of physicality in TUIs for introducing programming concepts to young children aged six-to-seven. A suitable time to introduce basic STEM knowledge and skills for young children is in the early elementary grades. This is because children at this age generally have the highest learning motivation, and are at their most eager to learn (Nadelson et al. 2013). In Saudi Arabia, first-grade students are aged 6 to 7 years old; this target population was used as the sample group for this study’s investigations.

Mixed results have been reported in studies comparing TUIs and GUIs for programming education based on collaboration support, attractiveness, programming achievement, enjoyability, usability and gender differentials (Horn et al. 2009; Kwon et al. 2012; Pugnali, Sullivan, and Bers 2017; Sapoundis and Demetriadis 2013; Strawhacker and Bers 2015). The previous studies have not identified the specific circumstances under which TUIs might offer more benefits for learning programming. Therefore, this research investigated the use of TUI as an input (programming environment and output engaging physical output) for learning programming. There is a need to investigate the critical differences in support in detail and examine learning outcomes.

One of the few experiments investigating the learning benefits of TUIs and GUIs in young children was Portelance et al.’s pilot study with 35 participants aged five-to-six years (Portelance, Strawhacker, and Bers 2016; Strawhacker and Bers 2015). The authors reported a possible association between the interface type and learning outcomes, but the result was not statistically significant. In addition, the study did not examine other learning benefits such as enjoyment.

Overall, little empirical work exists that provides evidence that TUI supports the learning of programming for young children (Bakker, Van Den Hoven, and Antle 2010; Marshall, Cheng, and Luckin 2010). Additionally, there is a lack of a theoretically grounded
framework that outlines how and why we might expect TUIs to mediate learning interactions and thus affect learning outcomes (Antle and Wise 2013).

1.4.2 Gestures and Conceptual Metaphor

There is a growing body of research on gestures in terms of their benefits in learning, mental modelling, reflection of understanding and reducing information processing and cognitive load. Schwartz and Black (1996) advocated that gestures are physical illustrations of mental models. In their study, they found that when solving interlocking gear problems, participants gestured the movement of the gears with their hands to support them in imagining the right direction of the gears. They then progressively learned to abstract the rule to solve the problem (Schwartz and Black 1996). In addition, Alibali et al. found that spontaneous gestures helped to externalise vital information about individuals' otherwise hidden mental representations of math-based problems (Martha W Alibali et al. 1999).

Researchers (e.g. (Goldin-Meadow 1999)) have suggested that gestures can provide knowledge that is not expressed in speech. It is possible to view gestures as a sign that the body is engaged when thinking and talking about the ideas. These gestures are considered evidence that the knowledge is embodied (Gibbs, 2006a; Hostetter & Alibali, 2008; McNeill, 2005; Núñez, 2005). According to the notion of embodied cognition, mental processes are mediated by body-based systems such as body form, mobility, and scale; motor systems such as the neural systems involved in action planning; and sensation and perception systems (Glenberg, Goldberg, and Zhu 2010; Suchman 1987). Therefore, gesture may activate action representation in brain and support memory by including more part of the brain. Gesture engages both the motor part and language part and activating multiple brain region may cause better learning (Cherdieu et al. 2017).

Even thought that evidence showed that physical manipulative can support learning in benefit learning in English (Glenberg et al. 2004) and science (Kontra et al. 2015), Gestures could influence cognitive simulation more than the actions by physical manipulative (Goldin-Meadow and Beilock 2010)
Even thought that evidence showed that physical manipulative can support learning in benefit learning in English (Glenberg et al. 2004) and science (Kontra et al. 2015), Gestures could influence cognitive simulation more than the actions by physical manipulative (Goldin-Meadow and Beilock 2010).

Goldin-Meadow found in 2009 that asking children to use gesturing while learning the new concept of the grouping strategy in mathematics enabled them to retain the knowledge and solve additional problems (Goldin-Meadow 2009). The author argued that gestures reflect thought and are early indicators of a change in the cognitive state. Thus, they can be used to identify the learning status of the learner.

In another study, Goldin-Meadow and colleagues found that children who were required to produce correct gestures learned more than children who were required to produce partially correct gestures, who in turn learned more than children who were required to produce no gestures during a mathematics lesson. As a result, the authors argued that bodily movements foster the understanding of old ideas and the creation of new ideas (Goldin-Meadow et al. 2009).

More recent evidence suggests that gestures could positively affect the structure of mental representations in mathematics-related disciplines, including problem-solving (Chu and Kita 2011). Gesturing can improve learning by reducing cognitive load (or cognitive offloading). For example, Cook et al. (2012) found that gesturing while performing elimination mathematical problems allowed speakers to recall more information than when they did not gesture (Cook, Yip, and Goldin-Meadow 2012).

CS is a discipline that relies on problem-solving skills that could potentially be taught more effectively by encouraging gesturing, as in mathematics. Overall, the question of which types of gesture could be more valuable for fostering understanding of a specific concept is significant in EC theory. In CS education, there is limited research that addresses the role of gesturing and hand movement. As a consequence, this research examines the effect of interface type (TUI or GUI) on children's use of spontaneous
gestures and of the production of spontaneous gestures on children's programming learning outcomes.

1.5 Research Contributions

The contributions of this thesis relate to the three embodiment approaches of physicality, gesture and conceptual metaphor. The contributions can be described under two different categories; (i) interaction design related contributions pertaining to the importance of physicality in programming environments for children aged six-to-seven years; and (ii) the understanding of the importance of gestures and conceptual metaphors in primary school CS education.

The findings of this thesis add to the literature in interaction design for CS education concerning the role of interface physicality in learning for primary education. Although it is commonly suggested that TUIs are better suited for primary education than GUIs, the findings in this thesis indicate that GUIs may be just as effective (if not more so) than TUIs. In particular, GUIs are associated with positive attitudes toward computing as well as enjoyment, which may be attributable to the long setup times for TUIs and the potentially distracting involvement of an adult in a TUI setting. This thesis also investigates the possible relationship between increased embodiment in interface (e.g., TUI), output device (physical robot) and increased use of embodied representations of concepts through gestures, finding no relationship across studies.

The thesis also contributes to the literature on the use of spontaneous gestures in CS education. First, the thesis describes and identifies children’s use of spontaneous gestures when solving programming tasks and describing programming concepts, finding that the participants mostly used pointing and literal gestures and few metaphorical gestures, which is expected at six-to-seven years old. The children were also observed using some similar metaphorical gestures as compared to adult learners when describing computing concepts. This thesis also provides evidence that children’s gestures can help characterise their conceptions in primary computing, potentially allowing identification of misconceptions and aiding the identification of productive pedagogic strategies. This
research also addresses the gap in the theoretical evidence base for understanding learning in CS. The work adopts an embodied cognition lens to promote understanding of the development of CS or potential approaches that draw upon embodied cognition to inform CS. Moreover, this thesis provides evidence that children use spontaneous gestures to demonstrate abstract computational concepts, even in the absence of relevant stimuli (i.e., written code). These gestures may reflect the embodiment of the children’s knowledge of computing concepts.

The findings regarding use of spontaneous gestures are broadly in keeping with those found in previous work on gestures in university-level CS students, but our findings demonstrate that the direction of written language in a culture may affect the direction/use of conceptual metaphor in CS. All previous work on the use of gestures in computing has been conducted with English speaking adults, no research has investigated the use of conceptual metaphor with Arabic speaking children. Furthermore, this thesis also reports a positive relationship between the mean learning gains of high-frequency gesturers and low-frequency gesturers when the tasks were characterised by varying problem difficulties.

1.6 Thesis Structure

Chapter 2 gives an overview of the literature related to the work presented in the thesis. The chapter first introduces the theory behind our empirical work, which is embodied cognition theory and its themes, including object manipulation and spontaneous gesturing. Second, it describes different types of learning interface used for teaching programming to students. Finally, it focuses on the literature that compares TUI and GUI and studies that explore spontaneous gesturing while learning computing.

Chapter 3 presents Study 1, which investigated a TUI and a GUI for controlling a physical robot (PR). The findings in answer to RQ1 indicated that that the GUI-PR yielded higher learning gains, while the TUI-PR improved attitudes towards computing. There was no difference between the interfaces in terms of enjoyment outcomes. The observations relating to RQ2 showed that frequent gesturers had higher learning gains than those who gestured less frequently. Additionally, this chapter describes children's spontaneous
gesture types, their location and purpose and the representational gestures that students made when referring to the notion of a loop.

Chapter 4 includes Study 2, which investigated a TUI-PR and a GUI-SR (with an on-screen robot) in terms of children's learning gains and their attitudes to computing, their enjoyment and their spontaneous gestures. It also discusses the relationship of children's learning outcomes and their spontaneous gestures. The findings in relation to RQ1 found that there was no significant difference between groups in terms of learning gains, while the GUI-SR group experienced a significantly greater improvement in attitudes toward computing. There was no difference between the groups in terms of enjoyment outcomes. In contrast to Study 1, the findings in regard to RQ2 found no relationship between the frequency of gesturing and learning gains. An analysis of the contrasting results is discussed. Additionally, the children's spontaneous gestures in Study 2 were similar to those in Study 1 in terms of their types and purpose, as were the representational gestures.

Chapter 5 presents Study 3 which was an exploratory study describing children's use of conceptual metaphors when they were describing programming concepts. The findings in relation to RQ3 were that the children produced less gestural evidence of metaphorical understanding compared to that of conceptual understanding. Additionally, Study 3 explored the potential roles of gestures as a tool to offload cognitive load and as a reliable means of communication instead of just using speech. This suggests that gestures might be an indication of the embodiment of the children’s computing notions.

Chapter 6 presents conclusions arising from this thesis, its limitations and directions for future work.

1.7 Related Publications


• Paper 3: “Investigating Primary School Children Embodied Expression of Programming Concepts.” This paper is submitted to and under revision at International Journal of Child-Computer Interaction (IJCCI).

1.8 Summary of Research Activities

The principal research activities during this research are shown in Table 1.1 below:

<table>
<thead>
<tr>
<th>Academic year</th>
<th>Achievement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Year 1</td>
<td>Participating at Doctoral consultum at the 32nd Human-Computer Interaction Conference (Almjally, 2018)</td>
</tr>
<tr>
<td>Year 2</td>
<td>Conducting Study 1</td>
</tr>
<tr>
<td>Year 3</td>
<td>Conducting Study 2 and Study 3</td>
</tr>
<tr>
<td></td>
<td>Published and presented a full paper to Special Interest Group Computer Science Education SIGCSE 2020. My slides <a href="https://drive.google.com/file/d/1hoxPDbgW5utD1d4Pke6uVsCPOUm6-">https://drive.google.com/file/d/1hoxPDbgW5utD1d4Pke6uVsCPOUm6-</a> edD/view</td>
</tr>
<tr>
<td></td>
<td>Published and presented of a full paper to Interaction Design and Children (IDC) 2020 in June 2020.</td>
</tr>
<tr>
<td>Year 4</td>
<td>Submitted article on Study 3 to the International Journal of Child-Computer Interaction</td>
</tr>
<tr>
<td></td>
<td>Writing up thesis</td>
</tr>
</tbody>
</table>
CHAPTER 2

EMBODIED APPROACHES TO LEARNING PROGRAMMING

2.1 INTRODUCTION

This chapter reviews the literature on primary computing education in the UK and the embodied approaches used to support learning fundamental programming concepts, e.g., tangible manipulation and spontaneous gestures. First, an overview is given of computing education in the Saudi Arabia. Second a deeper look on the computing education in the UK, including (1) enrolment numbers in post-compulsory-CS, (2) teachers' perspectives on CS teaching activities and (3) programming environments (e.g., GUIs and embodiment approaches including ‘unplugged’ types and TUIs) used at the primary level. Third, different views of embodied cognition are presented. A detailed examination is given of educational approaches based on embodied cognition (e.g., physical manipulation, image schemas, conceptual metaphors, gestures and their role in learning). Finally, a review is offered of empirical work that has evaluated the effect of physicality on learning fundamental programming concepts using block programming language by comparing TUIs versus GUIs for young children. Additionally, the chapter examines current work on the use of gestures in the CS domain.

2.2 COMPUTING IN PRIMARY SCHOOLS IN SAUDI ARABIA

Saudi Arabia is located in the Middle East. The official language is Arabic, and it is written from right to left which is in contrast to the direction of the programming block. The constitution and the law in Saudi Arabia derive from the Islamic laws which the Islam religion effect rules and regulations (Krivenko 2009). For example, genders are separated in education.

Typically, K-12 education and universities are gender-segregated and universities have gender-separated campuses. In some elementary school boys aged (6-9) can study in girls’ school which they learn by female teacher and the school administrations are female.
However, boys’ classes, interval time, and all other class activities are segregated from the girls (ARAB NEWS 2013).

In 1982, a Computing curriculum was introduced to secondary-boys school (ages 15-18) for the first time, for one hour per week. In 1993, the Computing curriculum started to be taught for two hours per week. In 1998, the research and library courses were merged into the CS curriculum and introduced a new version of Computing curriculum to intermediate level aged (12-15). In 2006, CS as a subject was considered a compulsory for all-female schools (Asharq Al-awsat 2006). The CS curriculum is the same for boys and girls (Al-Wakeel 2001). Currently, Computing is taught at intermediate and secondary schools but not at elementary level (age 6 to 12). The objectives of Computing curriculum in intermediate level are related to basic understanding of information technology, whereas the Computing curriculum for secondary level emphasizes programming, digital citizenship and contemporary applications.

2.3 Computing in Primary Schools in the UK

Despite the lack of a standard computing curriculum for primary schools in Saudi Arabia, there is a strong interest in devising and adopting such a curriculum. This overview is limited to the situation concerning computing education in the UK, which stems from the UK’s status as one of the first countries to introduce a compulsory computing curriculum for primary school students and the fact that this doctoral research was supervised at a UK University. Since September 2014, a computing course has been part of the national curriculum in England (GOV.UK 2013). However, computing education is still facing issues regarding the low enrolment rates at GCSE level (Kemp, Berry, and Wong 2016) and A-Level computing courses (Joint Council for Qualifications 2012). There is also a lack of (i) an agreed theoretical evidence base for computing education; (ii) empirical evidence of the usefulness of physicality; and (iii) the self-confidence of teachers teaching the subject. The next three subsections explore computing curriculum in the UK, students’ enrolment, teachers’ perspectives on CS teaching at the primary level and tools used to teach programming for children.
2.3.1 Curriculum in the UK

In September 2014, the Computing subject entered the national curriculum in England (“National curriculum in England: computing programs of study - GOV.UK,” 2013). Information and Communications Technology (ICT) was replaced by a Computing subject that includes but does not solely focus on programming. Instead, it includes Computer Science concepts explaining how computer works and how to give instructions to solve problems through programming. The course also employs information technology to create content, thus improving students’ digital literacy. The primary computing curriculum objectives are to ensure that all students:

- Understand and apply fundamental computer science ideas and concepts such as abstraction, logic, algorithms, and data representation
- Can analyse problems in computational terms and have repeated practical experience building computer programs to address such problems
- Analytically analyse and utilize information technology, especially new or unfamiliar technologies, to address challenges
- Become responsible, competent, self-assured, and innovative users of information and communication technologies

These objectives should be reached in four key stages covering primary and secondary education. Before deliberating about primary school computing key stages, a brief description of the UK education system is given. It is divided into four key stages as follows: Key Stage 1: 5 to 7 years old, Key Stage 2: 7 to 11 years old, Key Stage 3: 11 to 14 years old, and Key Stage 4: 14 to 16 years old.

Stage 1 is part of the early years of elementary education, Stage 2 is for the older elementary school, and Stage 3 and 4 are part of the secondary school curricula. This research is interested in the first year of elementary education; the other materials are out of its scope. The pupils in Key Stage 1 should understand what programs are and their use in programming on digital devices. Key Stage 1 comprises general guidelines. This research focuses on points A and B as they relate to programming:
A. Create and debug simple programs
B. Use logical reasoning to predict the behaviour of simple programs
C. Intentionally utilize technology to produce, organize, store, alter, and retrieve digital material
D. Recognize typical uses of information technology outside of school
E. Use technology responsibly and respectfully, keeping personal information private; know where to go for assistance and support if pupils have concerns about material or interaction on the internet or other online technologies.

These curriculum guideline goals are sustained by different teacher development support groups, such as computing at school (CAS), which is a community supporting computing in school. Its community in November 2017 had more than 28k members, 4277 computing resources, 91K discussion posts, and grew daily. Computers at school support teachers with workshops, materials, and local hubs comprising local and keen teachers who aim to share ideas about teaching computing. Barefoot (https://barefootcas.org.uk/) and Quickstart Computing are toolkit for primary and secondary teachers (http://quickstartcomputing.org), form part of the CAS group. Additionally keen computing teachers developed and shared teaching materials, as shown by the work of Phill Bagge through his website http://code-it.co.uk/.

2.3.2 Programming Concept for Age 6 to 7

Identifying the learning outcomes is challenging because few resources have identified the learning outcomes for children aged 6 to 7. Therefore, we used the UK Key Stage 1, from A to B, as a reference point, in addition to education (Robertson, Manches, and Pain 2017), computer science research (Futschek and Moschitz 2011), and computer education at organizations like Computing at School. All agree on the importance of teaching children from 7 years old the basic concepts of programming and simple algorithms in terms of their operation, creation, and debugging. Programming involves producing a set of instructions that tells a computer how to complete a task. Programming covers various concepts based on the language, but there are some common concepts that programmers use in any programming language. The basic commands that older primary students need
to understand, and use are sequencing, or running instructions in order, selection, or running a specific set of instructions depending on what happens, iteration, or running some instructions several times, and variables, which is a way of storing and retrieving data from the computer’s memory. However, there is disagreement about whether to teach selection in programming for children between 6 and 7. Futschek & Moschitz (2011) present the “Time the Train” learning scenario for primary school children, which allows different kinds of interactions with tangible objects for children to learn basic programming concepts. For example, selection and iteration are introduced together in the same activity, and children develop the loop step by step. First, children need to understand what a given algorithm does and then reformulate the given algorithm using the loop structure. Then, they need to know the conditions in which the algorithm is correct. This will help the children to understand the limitations of their solution. For example, in the loop activity, learners were asked to identify the repeated part of the algorithm. Then, they built the algorithm using a loop structure and identified the correct limit (stopping point of the loop), which included the condition or the alternative (Futschek and Moschitz 2011). However, Barefoot and Computing at School originally stated that their activity, for students from ages 5 to 7, has limited use of selection, as the program’s primary focus involves understanding sequences and repetition. Selection in programming is suitable for older primary students (Barefoot, n.d.). Therefore, we will focus on only two command types: sequence and iteration.

**ITERATION**

A sequence of code that is repeated until a specified condition is met or time elapses. In programming, iteration is often used because segments of code frequently need to be executed several times or until a certain condition is met. Therefore, iteration is important when learning any programming language because it simplifies the code by enabling the efficient repetition of instructions. Using iteration makes algorithms quicker and simpler because the number of irrelevant steps is reduced (Bitesize 2022).
2.3.3 Student Enrolment in Post-Compulsory CS

Computing has been a compulsory subject in K-12 education in the UK for approximately nine years and the number of schools offering computer science has since increased by 52.5% at GCSE level and by 36.2% at A-Level. Students will likely now find CS on offer at their school (76.3% of schools now offer CS at GCSE level). The main aim of introducing a computing curriculum in school was to increase the number of CS graduates. However, only 11.9% of students choose to take the subject at GCSE and only 2.7% at A-level (Peter, Kemp; Miles, Berry; Billy 2018).

The annual statistical report published by the Joint Council for Qualifications noted that computing had one of the lowest rates of female enrolment of any qualification at GCSE (16.1% female) and A-level (8.6% female). Female students are usually underrepresented in the CS domain. However, the report highlighted that girls outperformed boys in GCSE and A-level computing in the highest-grade bands (A*, A and B). Reviewing the literature on women’s experiences with STEM, including a reported avoidance of computing, articles reported that the number of females interested in pursuing computing drops at each level of education starting from elementary school to PhD (Camp 1997; Hope 2021). In order to improve attitudes toward computing, it is important to measure the attitudes towarded computing when children are programming so that we can better understand the opportunities and risks associated with different programming environments (Imhof, Vollmeyer, and Beierlein 2007).

Students’ attitudes toward computers in their early stages of learning about computing, particularly during their initial experiences of programming at the primary level, might have a major effect on their attitudes and decisions to choose CS for A-level or undergraduate learning (Kemp, Berry, and Wong 2016).

2.3.4 Primary School Teachers’ Perspectives

This section summarises teachers’ perspectives on teaching computing at primary level, identifying the challenges they face and the activities they view as useful for successful teaching. A UK-based study conducted in 2014 by Sentance et al. surveyed more than 300
computing teachers. The teachers reported that they used kinaesthetic activities (meaning interacting with physical materials), which are intended to promote understanding of a concept concretely and actively, to promote collaboration and computational thinking skills (Sentance and Csizmadia 2017, P485). The teachers believed generally that engaging physically (e.g. unplugged style activities) in computing learning classes was beneficial for students understanding, despite the absence of empirical evidence. Given the lack of empirical data, further exploration is needed (Sentance and Csizmadia 2017). Additionally, Sentance et al. surveyed teachers about the challenges associated with their computing knowledge, including those relating to programming, problem-solving and pedagogical approaches. They found that teachers often did not have extensive knowledge of computing themselves, which made it hard for them to support students in thinking through issues and breaking down and tracing problems rather than asking for support at the first indication of difficulty (Scherer, Siddiq, and Sánchez Viveros 2020).

Moreover, for some computing topics such as theory, some teachers perceived that these were complex to explain, while others described the task of engaging students as challenging. Like teachers, students also have challenges, which include linking basic concepts to practical applications. This challenge requires students to apply more than one basic idea to solve real-world problems (e.g., declaration and usage of a variable). Other broader difficulties that students face include problem-solving and engagement (Sentance and Csizmadia 2017).

2.3.5 Programming Environments in Primary Schools

This section describes the programming environments used in primary schools as either screen-based programming environments (GUIs on a desktop or touchscreen) or screen-free approaches such as CS Unplugged or TUIs and for each the embodied perspectives.

**Graphical User Interface (GUI)**

There are two main types of tools that have been used to teach and encourage students to learn to program: traditional text-based programming environments and block-structured programming environments. Text-based programming is mostly used in higher education,
GCSE and A-Level CS due to its high level of abstraction and the difficulty of learning the syntax. Alternatively, block-structured programming environments are commonly used in the initial stages of education to increase engagement with the subject and lower the barriers to entry. Examples of these programs include Scratch, Hopscotch, and Kodable (Hopscotch 2021; Kodable 2019; MIT 2003). Scratch remains the most widely used tool employed by UK primary schools to teach fundamental programming concepts.

Scratch aims to help non-technical users, including children, to implement their creative ideas using systematic reasoning. The user-friendly interface and the drag-and-drop programming blocks make Scratch easy to use for designing narratives (MIT 2003).

From this perspective of the notion of concrete to abstract in computing, the role of tangibles and metaphors is merely educational to support students in terms of abstract thinking, such as by offering visual imagery. Another example is the adoption of block programming environments as a visual metaphor to help children grasp programming language syntax.

‘UNPLUGGED’ APPROACH

Non-computing activities are available that leverage physical activity to engage children with body-based and physicality aspects of computing concepts (e.g., Unplugged), which offers a variety of non-computer-based activities and games (Bell, Lambert, and Marghitu 2012). CS Unplugged is a movement that encourages computer science learning without using computers, where the learning materials are based on physical movement and kinaesthetic activity (Garcia and Ginat 2012).

For example, Rescue Mission is a physical learning activity that places a child, ‘The bot’, and an object, ‘The bug’, on a physical grid in the classroom, or outside, on a painted chessboard. The aim of the learning activity is to ask the students in the class to write clear instructions for the bot to find the bug, which creates an opportunity to explore the concept of algorithms. Physical learning activities are effective because children learn through observations and experiences, as well as by being physically involved in the solution to a problem as it is being solved (Cortina 2015).
The teaching materials are based on interactive, collaborative and fun learning activities. However, CS Unplugged can only help in understanding programming concepts. It does not allow programs to be written and executed.

Further than motivation, Manches et al. (2020) stated that no clear framework exists that explains how body-based experiences may promote learning. Embodiment presents different stances on metaphors in computing cognition. Metaphors may be significant in the way we conceptualize some ideas in the domain of computing, rather than merely making abstract notions more tractable. Conceptual metaphors may be the techniques we use to understand, reason about, and convey computer concepts, which may have vital implications for computing education.

**TANGIBLE USER INTERFACE (TUI)**

The use of TUI for fostering programming skills is not a new idea (Perlman 1976). In 1976, Perlman presented a tangible programming language (TPL) to improve children's programming skills after observing their difficulties when they used Logo. The authors' solution consisted of two boxes: a button box, which had buttons to control a robot turtle and a slot machine which consisted of slots into which children could insert command cards. Children could then place a run card into the slot machine to run the commands in linear order. Following Perlman’s 1976 work, different TPLs have been introduced to foster programming skills in children, such as *OSMO* and *KIBO* (OSMO 2021; Pugnali, Sullivan, and Bers 2017).

TUIs give children the opportunity to directly manipulate tangible objects such as programming blocks to generate a program. Children can interact with such physical objects and transform the logic in of the physical world into the code logic. Additionally, TUIs are suitable for introducing young children to programming because they leverage children’s natural kinaesthetic learning strategies to make abstract concepts more accessible and intuitive.

In general, despite some evidence attesting to the benefits of TUIs in fostering programming knowledge and skills in young children, the available results are mixed (see
Section 2.6.1. Challenges exist in terms of the design, cost, manufacture, deployment, maintenance and updating of TUIs compared to other traditional interfaces, including GUIs (Shaer and Jacob 2009). Nevertheless, there is considerable interest in using TUIs in computing education, especially with children (Sapounidis, Stamelos, and Demetriadis 2016).

2.4 Embodied Cognition

Theories of embodied cognition highlight the extent to which the body and the physical world influence and are influenced by cognition, pointing to the coevolution of both body and behaviour, emotional states and culture (Glenberg, Goldberg, and Zhu 2010). The body and mind play important roles in cognitive development, and they are inextricable. Theories of embodied cognition are derived from the behavioural sciences, philosophy and ecological psychology, as well as other fields that describe people's behaviours and actions in the real world. There are different perspectives on embodied cognition (Semin and Smith 2008; Varela et al. 2016), including those of linguistics, cognitive science and HCI.

2.4.1 The Cognitive Linguistics Perspective on Embodied Cognition

The linguistic perspective highlights the grounding of semantics in bodily metaphors (Lakoff and Johnson 1999). Both in language and cognition, metaphors enable humans to understand one concept (target domain) in terms of another (source domain) (Bakker, Antle, and Van Den Hoven 2012). When the source domain of a metaphor involves schemata that have arisen from experiences relating to bodily movements, orientation in space, or its interaction with objects (Lakoff and Johnson 1999), it becomes a conceptual or embodied metaphor (Johnson 1987). Furthermore, we depend upon these embodied resources to think, reflect, reason, and communicate diverse ideas using conceptual mapping. For example, there is evidence that we use conceptual metaphors to comprehend abstract concepts in terms of concrete entities (Lakoff and Núñez 2000). Lakoff and Johnson (1999) distinguish, ontological and orientational metaphors. Ontological metaphors refer to the cases where an abstract concept is represented as concrete (e.g., an
object, container, person, or animal): for example, the mind is a machine. Orientational metaphors refer to the cases where concepts are spatially associated with each other (e.g., up-down, in-out) (Lakoff and Johnson 1999); for example, in language we can use the metaphor that happy is up, sad is down.

2.4.2 The Cognitive Science Perspective on Embodied Cognition

The cognitive science perspective addresses embodiment through the notion of embodied cognition, focusing on evidence suggesting that sensory representations and mental simulations can influence human learning, knowing and reasoning (Glenberg 1997; Pecher and Zwaan 2005). Grounded cognition advocates that a full understanding of a concept involves the ability to create a mental perceptual simulation of it when retrieving information or reasoning about it (Barsalou 2008, 2010; Black 2010). Results from behavioural and neuroimaging studies have shown that several psychological phenomena, initially thought to be absolute, symbolically demonstrate perceptual effects. For instance, property verification (e.g., retrieving the fact that a horse has a mane) was once thought to include a search from a concept node (horse) to a property node (mane) in a symbolic propositional network. Thus, the time to answer and errors were determined by the number of network links needed to be searched and the number of distracting links in existence. However, embodied cognition research shows that perceptual variables such as size influence verification times and errors (e.g., more important properties are retrieved faster) (K. O. Solomon and Barsalou 2004). A neuroimaging study using functional magnetic resonance imaging found that perceptual areas of the brain (responsible for shape, colour, size, sound and touch) become active during this task, not just the symbolic areas (Martin, 2007). Therefore, even for a person who knows about horses and manes, verifying this property involves a perceptual simulation.

Wilson (2002) discussed six distinct claims that are often argued for in the context of theories of embodied cognition, the first five of which are: (1) cognition is situated; (2) cognition is time-pressured; (3) we offload cognitive work onto the environment; (4) the environment is part of the cognitive system; and (5) cognition is for action (Wilson, 2002). These five claims are related to the way that humans use the environment as a dynamic
resource to reduce 'online' cognitive demands. The sixth claim states that even our 'offline thinking' (thinking that is decoupled from the external environment, such as planning or remembering) is grounded in mental structures that were originally evolved for interaction with the environment.

2.4.3 The Human-computer Interaction (HCI) Perspective on Embodiment

Embodiment correlates with the idea that human reasoning and behaviour connect with people’s bodies and their social and physical interactions with the world (Marshall and Hornecker 2013). The HCI perspective on embodiment represents an interactive relationship, whereby reasoning and behaviours can shape interaction and vice versa. Dourish introduced the term embodied interaction and defined it as ‘the creation, manipulation and sharing of meaning through engaged interaction with artifacts’ (Dourish 2001, p. 126). According to Dourish, employing artifacts such as tangible embodied with technology (TUI) in specific cultures, times, or environments could prove to be more suitable for users than traditional desktop computers. For instance, Educators can use physical manipulation to foster students’ learning. Underkoffler and Ishii’s developed urban planning workbench enabled users to flexibly explore interactions between wind, reflection and shadow effects for different configurations of buildings by manipulating tangible models (Underkoffler and Ishii 1999).

2.5 Educational Approaches Based on Embodied Cognition

Research is ongoing into the use of embodied cognition to understand learning and to determine how students’ progress from concrete to abstract concepts using different embodied concepts, including physical manipulation of objects (tangibles) and spontaneous gestures. First, there is evidence that physical manipulation supports the development of abstract reasoning, learning (Antle 2009), enhances understanding and improves memory recall (Antle, Droumeva, and Ha 2009). Antle (2009) showed that TUIs had a positive effect on problem-solving compared with GUI mouse-based input, which is consistent with theories of embodied cognition, arguing that the human body plays a significant role in thinking and acting in the world (Antle, Droumeva, and Ha 2009).
Second, research has shown that gestures can play a fundamental role in learning (Manches et al. 2020). For example, learners use gesture to express their ideas spontaneously (Novack and Goldin-Meadow 2015). Many researchers have studied motion and gesture to understand how people learn, including (Novack, Goldin-Meadow, and Woodward 2015) and (Manches et al. 2020). Educational research also shows that motor activity (e.g., physical manipulation or gesturing) supports retention of learned concepts because it establishes additional cues for representing and later recalling knowledge (Carbonneau, Marley, and Selig 2013; Chu and Kita 2011; Lee 2014). Rather than simply looking at or hearing information, engaging with supporting information and interactions promotes deeper processing that produces stronger memory traces, allowing learners to activate multiple memory retrieval avenues (Craik and Lockhart 1972). Furthermore, there is emerging research addressing how young learners progress from concrete to abstract representations of computing concepts (Trory, Howland, and Good 2018).

Computer programmers employ abstraction to conceal the material machine behind progressively complicated layers of code, layers that build a stack of abstractions, the lower ones hidden by the more complex layers on top. Students need an access point into the digital world of computing, and teachers need to facilitate it.

The following subsections discuss physical manipulation, image schemas, conceptual metaphors, and gestures in greater detail.

2.5.1 Physical Manipulation

Since the 1940s, the National Council of Teachers of Mathematics (NCTM) has encouraged the use of physical manipulation in students' mathematical learning activities (Learning Resources, n.d.). Researchers have found that physical manipulatives are a powerful addition to mathematics learning, with evidence to suggest that academic achievement can be increased by the long-term use of object manipulation (Moor 2012). Physical manipulation can help students to learn mathematical concepts by enabling them to establish connections between the concrete and the abstract (Heddens 1986). For
example, Schwartz and Martin found that when children used physical manipulatives, it facilitated their ability to develop their interpretation of fraction (Schwartz and Martin 2006).

Aggarwal et al. completed an empirical study investigating the impact of physical manipulatives on the ability of 8-to-11-year-old students to understand, recognise, construct and use a game programming tool called Kodu. They found that the students who used physical manipulatives performed better in rule construction compared to the students who did not use manipulatives. The reason for this result seemed to be that the students who used manipulatives before implementing the activities in the programming editor (in this case, Kodu) might have developed a more refined understanding of Kodu's rule syntax (Aggarwal, Gardner-McCune, and Touretzky 2017).

The act of physically manipulating an object is different compared to interacting with other types of interfaces, such as GUIs. Physical actions made by using manipulatives can support learning (Gravemeijer 1994; McNeil and Jarvin 2007). The literature on embodied cognition suggests that knowledge is grounded in perceptual experience and not abstract. This implicates a greater role of tangibles in promoting experiences for developing concepts. It entails creating spatial hand movements and influencing how objects are moved. For example, (Manches and Price 2011) distinguished between the actions produced by the hands to examine the role of physical manipulation in learning. They investigated the cognitive benefits of physical manipulation and reported that this may aid learning by creating actions that allow children to draw on prior information or by generating essential motoric representations to assist other types of representation. The authors also found that physically manipulating representations may even enable students to learn by allowing them to easily explore a variety of representational states, which limits their behaviours as they create new concepts.

### 2.5.2 Image Schemas and Action

Another method that enhances learners' understanding of the connection between their bodily interactions and their surroundings is related to the role of image schemas in
expanding thinking (Antle 2009). According to embodied cognition theories, image schemas are abstract mental structures that are created over time from repeated patterns of experience in the world (Johnson, 1987) and thus inform our understanding of new experiences. Lakens et al. (2011) demonstrated how individuals use spatial distance (e.g., near versus far) to examine and communicate differences between concepts that illustrate the use of image schemas as scaffolds in a brain classification task. In their study, they asked participants to perform two key-press versions of the Stroop colour interference task. The task revealed that the subjects responded quicker to words written in a specific colour (e.g., red) than when the meaning was congruent with the colour (e.g., the word red written in red). The subjects also responded quicker to natural phrases (e.g., the letter string XXX written in red) than words where the meaning did not align with the colour (e.g., the word blue written in red). The study employed a two-colour version of the Stroop task in which participants made a manual binary categorisation of the red and blue stimuli. The response keys were placed far apart during one block of the task and close together during the other. Correct responses on incongruent trials in the Stroop task required more cognitive control than correct responses on congruent or natural trials.

The author found that responses on incongruent Stroop tasks were quicker with the response keys placed far apart instead of close together (Lakens et al. 2011). The spatial structuring of response options facilitates classifications that require cognitive effort and incorporating environmental structures such as spatial distance into thought processes (Clark 2008)

Therefore, there is evidence in the physical manipulatives domain that leveraging image schemas in input actions can improve usability and system learnability (Antle, Droumeva, and Corness 2008; Bakker, Van Den Hoven, and Antle 2010).

2.5.3 Conceptual Metaphors

Conceptual metaphors represent embodied experiences as they involve the production of image schemas that map metaphors to abstract concepts. For example, the concept of time is grounded in the experience of linear motion and in many cultures, the past is ‘behind’
while the future is ‘ahead’ (Núñez and Sweetser 2006). A conceptual metaphor involves a mapping between a source domain (e.g., movement) and a target domain (e.g., time). The target domain is understood unconsciously in terms of the relations that are rooted in the source domain, which is entrenched in everyday sensorimotor experience. Thus, conceptual metaphors demonstrate the human ability to think and reason about abstract concepts (Kövecses 2008). Most empirical work investigates conceptual metaphors deriving from language. For example, in mathematics, words such as ‘count out’ and ‘next’ might indicate a conceptualisation of numbers as a collection of objects or as points along a path (Lakoff and Núñez 2000). Language is not the only way people use conceptual metaphors; they can also be expressed using gestures, which provide another way to express our understanding of abstraction.

2.5.4 Gestures

Gestures produced when speaking are a pervasive element of human communication and can reflect what people are thinking (Roth 2001). Three types of gesturing are discussed in the literature on mathematical learning (Broaders et al. 2007; Cook, Yip, and Goldin-Meadow 2012; Novack et al. 2014) namely:

- Deictic gesturing (e.g., pointing): This reflects the grounding of cognitive gestures in the physical environment (e.g., pointing to a pen to refer to it specifically).

- Representational gesturing: Includes gesturing to represent concrete or abstract concepts by hand (e.g., hand shape or motion trajectory) or body (e.g., rotating the whole body to indicate a turn) (Martha W. Alibali and Nathan 2012). A representational gesture may be either:
  
  o Literal (i.e., iconic gesture): The gesture demonstrates concrete objects or actions (e.g., depicting the act of writing); or
  
  o Metaphorical: The gesture resembles a familiar object to represent an abstract concept (e.g., cupping hands as if they were ‘holding’ an idea, which indicate the metaphor of idea as an object).
Gestures allow humans to construct complex explanations by reducing information processing and cognitive load (Schwartz and Black 1996). The following subsections explore the benefits of gestures in supporting learning and teaching. Most prior studies have explored the role of gestures in mathematics learning, which is relevant to consider given that the subject is closely related to CS. However, studies in the specific field of computing education are limited.

2.6 Roles of Gestures in Learning

Gestures may have more than one role in learning. For example, gestures may be simply indicative of understanding, they may be used to convey understanding, or actively help in processing and learning, including problem-solving and/or generating new ideas. Distinguishing between these different roles in some contexts can be very difficult.

2.6.1 Enhancing Problem-solving and Reducing Cognitive Load

Research interest has been growing concerning spontaneous gestures and their benefits in learning, mental modelling, reflection of understanding and reducing both information processing and cognitive load. Schwartz et al. advocated the notion that spontaneous gestures are physical illustrations of mental models. In their study, they found that the participants gestured with their hands to represent the movement of gears when solving interlocking gear problems, which supported them in imagining the correct directions of the gears. They then progressively learned to abstract rules over mental depictions to solve gear problems (Schwartz and Black 1996).

Cook, et al. found that performing gestures that coordinated with the speech while solving math problems enabled the participants to recall more information compared to when they did not gesture (Cook, Yip, and Goldin-Meadow 2012). There is also evidence that unprompted gestures have a beneficial impact on learning. For example, Chu and Kita (2011) found that learners who faced difficulties when solving spatial visualisation problems spontaneously produced gestures to help with problem-solving and that this had a positive effect on their performance. In addition, it is notable that the benefit of gestures appears to be distinct from the benefit of physical actions (Chu and Kita 2011). Novack
et al. compared physical actions performed on objects with gestures (Novack et al. 2014). They found that children trained to gesture performed better on near- and far-transfer problems compared to children who were trained to perform physical actions on objects.

2.6.2 Generating New Ideas

Gestures can help students generate new ideas and enhance their learning. Broaders et al. found that students who were asked to gesture when solving mathematics problems produced more correct and novel strategies when finding a solution than students who were not asked to gesture. Additionally, children who were asked to gesture achieved higher learning gains than children who were not asked to gesture (Broaders et al. 2007).

Goldin-Meadow et al. conducted a study in which children were randomly allocated into three groups and given the same equation to solve: $6+3+4=\_+4$. In the no-gesture group, children learned the phrase, ‘I want to make one side equal to the other side.’ The correct gesture group also learned the phrase and used gestures to illustrate, ‘V-hand to show $6+3$, point with the index finger to the blank’. The partially correct gesture group also learned the phrase but used different gestures to illustrate, ‘V-hand pointing at $3+4$, index finger pointing to the blank’. The V-hand indicated that two numbers can be summed together in conjunction with the point at the blank. Such a gesture emphasises the fact that the equation has two sides. The use of gestures also highlighted that applying them to two numbers can solve the problem (Goldin-Meadow, Cook, and Mitchell 2009).

All children practiced the words or the gestures they had learned before the researcher gave them a new problem to solve: $5+6+3=\_+3$. The children had to fill in the correct answer, then explain how they solved it without using gestures. Then, the children were given one new problem and asked to solve it using the gesture or no-gesture strategy. This alternating process took place 12 times, with the researcher and the children solving six problems each. The study found that children who produced correct gestures learned more than those who produced partially correct or no gestures. Additionally, their results suggested that children's ability to gain information only through gestures and to integrate this information into their communication grew, thus reinforcing the value of using body
movements to foster an understanding of old ideas as well as the creation of new ideas (Goldin-Meadow, Cook, and Mitchell 2009).

It has been suggested that gesture can offload cognition because it may help to establish and reflect imagistic representations activated while speaking (De Bot and Schrauf 2009) which correspond to prior integration with physical material (Roth 2002). For example, it has been reported in the literature that when teachers are asked to explain the concept of a fraction, they frequently use gestures related to actions with physical materials (Edwards 2009b). The available evidence indicates that gestures play a key role in supporting human cognition communication (Iverson and Goldin-Meadow 1997; Kelly et al. 2002). Even though certain gestures mirror earlier acts with physical materials, it does not follow that encouraging these sorts of activities using tangibles would result in learning advances (Almjally, Howland, and Good 2020b). However, there is evidence that asking people to use gestures can help them to offload the cognitive burden arising from a task (Goldin-Meadow et al. 2001), and facilitate the recall of implicit knowledge (Broaders et al. 2007). Limited work in the literature that investigated whether the use of tangibles activated or influence gesturing (Manches and Price 2011).

2.6.3 Revealing Implicit Knowledge

Gestures can convey knowledge that is not expressed in speech either incidentally or deliberately to assist the observer’s understanding (Goldin-Meadow 1999). Therefore, understanding gestures can provide teachers with insights into learners' thinking (Novack and Goldin-Meadow 2015). For example, after presenting a Piagetian conservation problem to a child participant in their research, Novak and Goldin-Meadow described how the child stated that a given volume of water changed when it was poured from a tall, thin container into a short, fat container. This indicated that the child did not understand the concept of conservation. The child justified their belief by saying, ‘This one is taller than this one’, while producing a C-shaped gesture representing the narrow width of the tall container, followed by a wider C-shaped gesture representing the wider width of the short container (Novack and Goldin-Meadow 2015). Children's gestures can convey knowledge that is not expressed in speech (Goldin-Meadow 1999) and understanding these gestures
can therefore provide teachers with insights into learners' thinking (Novack and Goldin-Meadow 2015). Alibali et al. found that spontaneous gestures externalised vital information about an individual's mental representations of math-based problems (Martha W Alibali et al. 1999). Thus, in a teaching and learning context, they can be used to identify a learner's understanding.

Researchers such as Goldin-Meadow have reported in other studies that gestures can provide an indication of knowledge that is not expressed in speech (Goldin-Meadow 1999). Goldin-Meadow found that asking children to use gestures while learning a new mathematical concept (in this case, the grouping strategy) enabled them to retain the knowledge and solve additional problems (Goldin-Meadow 2009). The researcher suggested that gestures reflect thinking and serve as early indicators of a change in cognitive state.

Bakker et al. conducted a user study with 65 children aged seven-to-nine years old to identify conceptual metaphors by asking children to enact abstract concepts related to musical sound. They found that the multiple different embodied metaphors can unconsciously be used to structure the understanding of these concepts. The result showed that many metaphors with incorrect verbal explanations were produced, an indication that knowledge acquisition takes place when shifting between experiences and reflections. The researchers argued that abstract concepts are first understood in terms of the experience before they can be explained in speech. On the other hand, the low number of metaphors they observed may be connected to their difficulty for the age group (Bakker, Antle, and Van Den Hoven 2009).

2.6.4 Effect of Teacher Gesturing on Learning

There is evidence that when teachers use gestures, students can benefit more from instruction (Church, Ayman-Nolley, and Mahootian 2004; Ping and Goldin-Meadow 2008; Valenzeno, Alibali, and Klatzky 2003). The literature suggests that co-speech gestures are powerful in communication and learning.
Valenzeno et al. investigated the effect of teachers’ gestures on children’s comprehension and learning. They viewed one of two videotaped lessons about the concept of symmetry. In the verbal-plus-gesture lesson, the teacher pointed to and traced the video as she explained the concept. In the verbal-only lesson, the teacher did not use any gestures during the video. After the session, children were given a post-test and asked to judge six items as symmetrical or asymmetrical and clarify their judgments. The gesture group scored statistically higher on the post-test than children who saw the verbal-only lesson. This result suggests that pointing and trace gestures can ground teachers’ speech by linking abstract, verbal utterances to the concrete, physical environment to facilitate student learning (Valenzeno, Alibali, and Klatzky 2003). This finding of Valenzeno is in line with Church et al.’s finding that children who watched video recordings in which the teacher conveyed instruction through a combination of speech and redundant gestures were significantly more likely to learn mathematical concepts and retain the new knowledge (Church 1999). Additionally, Ping et al. (2008) found that children taught using gestures outperformed their counterparts on solving Piagetian conservation problems, regardless of the physical presence of objects during instruction (Ping and Goldin-Meadow 2008).

There is also evidence that gestures can have a positive impact on learning even when the accompanying spoken language is not familiar to the learner. A study conducted by Church et al. examined the role of gestures when speech was not understood by the learner. They tested 51 children aged six-to-seven, of whom 26 spoke only English and 25 spoke only Spanish. Half of the English and half of Spanish speakers viewed a maths lesson in English, while the other half viewed a maths lesson in English together with gestures. They found that learning increased for all students when gestures accompanied speech instruction. Notably, Spanish speakers’ learning increased by 50%. They argued that gestures are not tied to a specific language, reflecting concepts in the form of universal representation. Another explanation for this finding could be that the gestures drew the attention of the learners to important issues in the maths (Church, Ayman-Nolley, and Mahootian 2004). This research suggests that gesturing might facilitate the grounding of
a lesson’s abstract language in the concrete physical environment, conveying ideas through a representational form (Novack, Goldin-Meadow, and Woodward 2015; Ping and Goldin-Meadow 2008) or it might just help make a link between the maths representation and speech.

In conclusion, there is extensive evidence that gestures play helpful and varied roles for both learners and teachers in a variety of educational contexts.

**2.7 Embodiment and Computing**

Conceptual development in computing can be understood through the lens of EC theory. First, embodiment supports how learners use their body-based experiences to make the digital analogy that can lead to implications for understanding how and through what mechanisms learners understand computing.

In other domains of STEM education there is evidence that learners frequently offload their cognitive load using embodied representations. Embodiment provides learners a way in to configure and think with the abstract (Weisberg and Newcombe 2017). For example, Enyedy and colleagues investigated the effect of physical activity on student slipping on linoleum, which later helped them explain how the speed of an object increases (Enyedy, Danish, and DeLiema 2013). There is a body of research which has suggested that embodiment is educational; it gives a way to understand computing concepts (Cortina 2015).

Other researchers (E. W. Dijkstra and Parnas 1989; Simon 1996) disagree about the effectiveness of taking an embodiment view of CS learning. Computing is not visual, as equivalent to other STEM disciplines, in the sense that learners cannot see bits, bytes, or loops. Computing is a science of the artificial, which humans have designed, and it is not inherited from nature (Simon 1996). Dijkstra and Parnas (1989) argued that computing is novel and cannot be represented by analogies or metaphors. Therefore, this implies that students cannot be taught using embodiment approaches (e.g. physical blocks and gestures) (E. W. Dijkstra and Parnas 1989).
However, others believe that the role of embodiment in learning does have implications for disciplines such as computing, which is based on abstraction and is virtual in nature (Stevens 2011). Using the sense of our body movement was identified as contributing to learning Logo because it helps learners to make sense of abstract issues using concrete examples (Papert 1980). In addition, Papert suggested that embodiment for learning CS is not only a problem-solving strategy, but grounds the abstract, and forms a representation that students can then think with and think through. Papert also noticed that children identified with the robot they were programming by linking their knowledge about their body movement into the work of learning formal programming. This suggested that embodiment for learning computing is not only a problem-solving strategy, but also grounds the abstract and helps to form a representation that learners can then think with and think through.

Abstraction is acknowledged to be a key skill in thinking about computing and computational concepts (Wing 2008), but it can be challenging for younger children. There is evidence that computing concepts can be progressed from embodied to abstract, including from recent research looking at how primary school students progress from concrete to abstract representations of computing concepts (Trory, Howland, and Good 2018).

Another reason for using embodiment representation in learning CS is that the focus is not only related to the learning outcomes and what is happening in the brain, instead it considers the whole learning context, which allows the researchers to investigate the practices and sociocultural contexts (e.g. attitude and enjoyment). These aspects help learners with understanding and reasoning. Investigating practices has direct implications for education; this process gives us the opportunity to explore how we might design learning environments that improve students’ learning experiences.

Beside the role of embodied representations in education to support students’ learning, embodied representations such as metaphors allow conceptualizing computing concepts (Manches et al. 2020). Manches and colleagues (2020) argue that metaphors underpin the
way people think and conceptualise about computing and argue that the concept of ‘abstract’ concepts in computing is problematic.

This thesis uses embodiment as a theoretical framework to document the ways in which embodied representations (i.e., physical manipulation, gestures, and metaphors) arise when children solve programming tasks or explain programming concepts. The implication is that humans understand the world and acquire new knowledge, skills, and abilities through thinking and learning from their bodies, as well as their own experiences, and not only through their minds. The aim of this PhD research is to investigate, analyse, and describe the embodied representations used by children when solving programming problem, and explaining abstract concepts.

2.7.1 Embodiment and Learning Programming

Programming is a foundational concept in CS, but learning to program is not easy (Sorva 2013). Even the most basic programming concepts, which are elementary to specialists, are commonly studied among novices and found difficult to overcome (Sorva 2013). Programming concepts are precisely defined and implemented, and students are required to understand what certain constructs and concepts do, such as variables, variable assignments, and flow of execution (Arawjo 2020; Boulay 1995).

Programming languages, on the other hand, are cultural instruments with complex syntaxes that were often conceived and constructed for professional usage, rather than to facilitate learning (Arawjo 2020). Learning programming is not easy because students find it difficult to understand; they do not grasp the program's critical features and do not know how to handle them by writing the program. (Boulay 1995). Students find it hard to create a mental model of how computing concepts work, a prominent example of which is the concept of pointers (DeLiema and Francis 2013). Responses such as this make iteration difficult to learn for students at the primary level. Abstraction in computing is ultimately about hiding information. Programmers write lines of code to instruct the compiler, and these lines of code do not demonstrate the complex computation and operations that are happening behind the scenes (Colburn and Shute 2008).
2.7.2 Blocks Domain Analysis

Although CS is a man-made discipline that is not inherited from nature and is not visible, spatial thinking may nonetheless influence learning CS (Cooper et al. 2015; Parkinson and Cutts 2018). The way in which teachers have been observed to use visual representations in CS classrooms indicates that visualizations are vital in CS education; this is because they create something tractable for students to develop conceptual understandings with (Larkin and Simon 1987). Teachers have been reported as believing that engaging students physically in learning activities is conducive to their success, despite the absence of empirical evidence (Scherer, Siddiq, and Sánchez Viveros 2020).

While physical representations, such as programming blocks, are frequently used in computing education, their physical character has some drawbacks. Traditional programming does not give haptic information as feedback, has no account of previous interactions, and requires explicit linkages to other representations. Computer-based representations may solve a variety of these restrictions, which helps to explain the prevalence of virtual representations in study and practice. However, physical programming blocks are mostly used in school, and in the literature, there is mixed evidence about their potential benefits.

In the action of switching and linking blocks, a child generates more frequent and greater movements in their hands and arms, resulting in the creation of a program. However, with the GUI, children tend to generate similar actions in terms of moving and linking blocks, but this is accompanied by limited hand and arm movement. Both the TUI and GUI in programming require some hand movement, but they are different in terms of the type, amount, and directness needed. The TUI enables children to touch and feel the blocks, which allows further movement by moving the whole arm to grip the tangible blocks, to follow the physical robot, and to turn the block in the required direction.
2.8 Empirical Work Investigating the Effect of Physicality (TUI and GUI) and Gestures

The following subsection describes empirical research investigating the use of different types of programming languages, namely, TUI and GUI, for teaching fundamental programming concepts to children. Subsection 2.6.2 addresses the few studies that investigate gesture use in the computing domain.

2.8.1 Comparing TUIs and GUIs for Children's Programming

This section addresses prior studies that have compared the use of TUIs and GUIs in young children to learn programming. Each of the existing studies examined different features, which are summarised in Table 2.1.

<table>
<thead>
<tr>
<th>Factors investigated and reported</th>
<th>(Horn et al. 2009)</th>
<th>(Kwon et al. 2012)</th>
<th>(Sapountzis and Demetriadis 2013)</th>
<th>(Sapountzis, Demetriadis and Stamkos 2015)</th>
<th>(Strawhacker and Bers 2015)</th>
<th>(Puglisi, Sullivan and Bers 2017)</th>
<th>This doctoral research</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age</td>
<td>4-8</td>
<td>6-7</td>
<td>Multi</td>
<td>Multi</td>
<td>5+</td>
<td>-7</td>
<td>6-7</td>
</tr>
<tr>
<td>Debugging</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Programming achievement</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learning outcomes</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enjoyment</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Collaborative</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Usability</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time spent</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Engagement</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Attitude</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gender</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Error</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Compared with identical GUI</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
Table 2.1 presents a summary of prior studies that compared the use of TUIs and GUIs for learning programming with children. The table excludes commercially produced evaluations of products such as *Osmo*, *littleBits*, and *Cubetto* (Cubetto 2021; Ofer et al. 2018; OSMO 2021), as well as tools that have not been evaluated empirically, including *Dr. Wagon* (Chawla et al. 2013).

Mixed results have been reported in studies comparing TUIs and GUIs for programming education which have evaluated differences in collaboration, attractiveness, programming achievement, enjoyment, usability and gender differentials (Sapounidis and Demetriadis 2017)

Horn et al. conducted a non-school study in a museum, which involved interviewing 13 family groups and observing 152 adults and 104 children using TUIs and GUIs (Horn et al. 2009). The study was between-groups using similar GUI and TUI interfaces. The results indicated that both interface types were equally easy to use, and no substantial difference was identified in terms of user performance or learning gains. However, the TUI was found to be more inviting, supportive of collaboration, child-focused and appealing to children under 16 and it had a more positive effect during the initial stages of learning. This result is consistent with similar results reported in other contexts (Cheng et al. 2011; Kwon et al. 2012).

Similar to Horn et al. (2009), Kwon et al. (2012) compared TUI (Algorithmic Bricks) to GUI (Scratch) in terms of usability, enjoyment, error count and achievement. The results showed no significant differences in terms of usability, enjoyment and level of achievement. However, error count for logical error (which includes errors in (i) setting the direction of the robot, (ii) the value of parameters and (iii) conditional statements for each of the programming tools) was significantly lower in the TUI group that suggested the effectiveness of expressed logical thinking through the use of tangibles. Additionally, the requests for help were counted when students had asked teachers questions about a problem, the low number of such requests in the TUI group suggested that TUI group

| Number of spontaneous gestures | | | | | X |

|  |  |  |  |  |  |

Table 2.1 presents a summary of prior studies that compared the use of TUIs and GUIs for learning programming with children. The table excludes commercially produced evaluations of products such as *Osmo*, *littleBits*, and *Cubetto* (Cubetto 2021; Ofer et al. 2018; OSMO 2021), as well as tools that have not been evaluated empirically, including *Dr. Wagon* (Chawla et al. 2013).

Mixed results have been reported in studies comparing TUIs and GUIs for programming education which have evaluated differences in collaboration, attractiveness, programming achievement, enjoyment, usability and gender differentials (Sapounidis and Demetriadis 2017)

Horn et al. conducted a non-school study in a museum, which involved interviewing 13 family groups and observing 152 adults and 104 children using TUIs and GUIs (Horn et al. 2009). The study was between-groups using similar GUI and TUI interfaces. The results indicated that both interface types were equally easy to use, and no substantial difference was identified in terms of user performance or learning gains. However, the TUI was found to be more inviting, supportive of collaboration, child-focused and appealing to children under 16 and it had a more positive effect during the initial stages of learning. This result is consistent with similar results reported in other contexts (Cheng et al. 2011; Kwon et al. 2012).

Similar to Horn et al. (2009), Kwon et al. (2012) compared TUI (Algorithmic Bricks) to GUI (Scratch) in terms of usability, enjoyment, error count and achievement. The results showed no significant differences in terms of usability, enjoyment and level of achievement. However, error count for logical error (which includes errors in (i) setting the direction of the robot, (ii) the value of parameters and (iii) conditional statements for each of the programming tools) was significantly lower in the TUI group that suggested the effectiveness of expressed logical thinking through the use of tangibles. Additionally, the requests for help were counted when students had asked teachers questions about a problem, the low number of such requests in the TUI group suggested that TUI group
were more able to solve problems independently than GUI group. Additionally, the TUI in this study may have had a positive impact on the earlier stages of programming level when children learned how to solve problems and use programming tools. Although the achievement level difference was not statistically significant, task achievement scores for the TUI students were marginally higher than the GUI students.

Sapounidis and Demetriadis conducted a study with several age groups, including five-to-six years, seven-to-eight years and 11-12 years, to explore their preferences, enjoyment and ease of use when using a TUI (*T_ProRob*) and a GUI to program a physical robot (Sapounidis and Demetriadis 2013). The researchers collected data via questionnaires, observations and interviews. The children in the three age groups were divided into two experimental groups according to their preference for which environment to use to start programming the robot and a presentation of how to use the two systems was given. The children tried one system and accomplished two tasks and later they swapped to the other system and performed similar tasks. In turn, the children had a free interaction session in which they could program the robot twice using each system successively and they then completed surveys about their enjoyment and perceived ease of use. The results showed that both interfaces were equally easy to use, except that children aged 11-12 years found the GUI easier. All groups found the TUI more enjoyable and it was identified as more attractive but not statistically significantly so. Girls identified the TUI as more attractive than the GUI and the difference was statistically significant.

Sapounidis and Demetriadis conducted another similar study in a school with 109 children in several age groups: six-to-seven, seven-to-eight, nine-to-ten, 10-11 and 11-12 years (Sapounidis, Demetriadis, and Stamelos 2015). The researchers measured three variables related to the children's task performance: time to accomplish tasks, number of erroneous programs and the number of debugging stages required after a programming error. The researchers also measured four variables associated with performance during free interaction with the TUI and GUI interfaces: program length, program vocabulary, program complexity and free interaction time-engagement (i.e., the time that children spent freely interacting with systems to create programs). Data was collected using video—
audio recordings and system logs. The results showed that the TUI users had lower error rates and engaged in more effective debugging and children took less time to complete a simple sequence-programming task using the TUI. In free interaction, older children in the GUI group used only a few parameters and commands to explore the abilities of the robot widely. In contrast, their counterparts in the TUI group used many parameters and commands.

None of the above studies examined learning outcomes or how the variable of interface type influences programming strategy.

One of the few experiments investigating the learning benefits of TUIs and GUIs in young children was conducted by Strawhacker et al. (2013). They examined three interfaces, a TUI, a GUI and a hybrid user interface (HUI) for controlling robots designed to support children in learning fundamental programming concepts (Strawhacker, Sullivan, and Bers 2013). They found that the TUI outperformed both groups in the easy and hard ‘sequence’ assessments, easy ‘iteration’ assessments and the culminating assessment. Nevertheless, the GUI group performed the best in the difficult ‘iteration’ assessment. The HUI averages did not exceed the averages of the other two groups and overall, the results did not favour one of the interface types. However, the TUI group had significantly higher scores for the easy sequence tasks than GUI group. It seems that the TUI group may have been able to dedicate more attention and cognitive resources to understanding the relationship between the programming blocks and the robot movements. The manipulability of the TUI may have helped children mentally model abstract programming ideas and the more time spent using tangible interfaces, the more they mastered programming. However, the groups in this study were exposed to different classroom cultures, as the GUI group came from Montessori-style classrooms and the others from more traditional educational cultures. Additionally, 15% of the students in the sample had a type of disability which the researchers could not distinguish due to of the school policy of not revealing this information. All groups had students with varying degrees and types of learning disabilities but, the specific individualities who has the disability, were not revealed to the researchers.
Another study, conducted by Pugnali et al., explored the affordances of two very different programming interfaces for teaching programming *ScratchJr*, a GUI on an *iPad* and TUI using a robotics kit (KIBO). The number of participants was 28 and they were aged four-to-seven years old. The authors reported an association between interface type and learning outcomes, but the result was not statistically significant. In addition, the study did not examine other learning benefits such as collaboration or engagement (Pugnali, Sullivan, and Bers 2017).

Rose et al. explored young children's programming approaches are using two GUIs: *ScratchJr* and *Lightbot* (Rose, Habgood, and Jay 2017). They used a non-verbal reasoning test to perform a matched assignment of 40 participants aged six-to-seven years under the two conditions. They identified two approaches, ‘top-down’ and ‘bottom-up’, following Turkle and Papert’s 1992 identification of two different approaches to programming: ‘the first is a top-down approach, where the solution to a problem is planned; the second is a bottom-up approach, which is attempted by arranging and rearranging the problem materials (Turkle and Papert 1991). They reported that both groups performed similarly overall, but the high performers using *ScratchJr* performed more program manipulation or ‘tinkering’, suggesting that they used a bottom-up approach.

Gender preference results have so far been inconclusive. Some studies have found that TUIs appear to attract both girls and boys equally (Horn et al. 2009), while other evidence indicates that GUIs appeal more to young boys (Strawhacker and Bers 2015). Another study reported that girls considered TUIs more attractive (Sapounidis and Demetriadis 2013). Therefore, more investigation is needed of the influence of gender on children's interface preferences and their ability to learn computing concepts.

Overall, most of the studies did not explicitly address the role of physicality in learning and they neglected to explore how interfaces influence different outcomes (e.g., learning outcomes or attitudes toward computing). Additionally, most of these studies did not use post- and pre-tests to measure learning outcomes and they tended not to include control
groups. This kind of investigation could be achieved in an experimental setting, enabling a between-group study to be undertaken.

2.8.2 Exploring the Role of Gestures in Learning Computer Science

Gestures have been studied in diverse fields within educational studies, including mathematics learning and reading comprehension (Martha W. Alibali and Nathan 2012; Cook and Goldin-Meadow 2006). However, there is limited research on the role of gestures in computing education. In this section, an overview is given of the few studies that have explored the role of gestures in computing education.

Solomon et al. conducted an exploratory study observing high school students (n = 8) and their use of gestures as part of a 12-week introductory computing course using Scratch. The students learned about variables, conditional statements and loops. At the end of the course, 15-minute interviews were conducted with the participants in which they explained the topics they had covered in the course. The research aim was to understand the types of gestures that the students and teachers produced in the course. The authors used McNeil's textonym, a gesture taxonomy for mathematics and they reported on some of the challenges of fitting McNeil's taxonomies to the data, emphasising the need for a particular conceptual framework to support gesture analysis in computing education (McNeill 1994). They stated that gestures in computing courses are potentially used as problem-solving strategies and as a way to communicate students' understanding of abstract concepts (A. Solomon et al. 2018).

A more recent study by Manches et al. investigated the conceptual metaphors generated by university-level computing students (Manches et al. 2020). They examined gestures by asking the 16 participants to explain three computing concepts (algorithm, loop and conditional). Although the students were not asked to use gestures, the participants produced 368 representational and metaphorical gestures, suggesting that explanations of CS concepts are often embodied through gestures. They found that participants mapped their gesturing onto two embodied metaphors. The first metaphor was ‘computing as a physical object’. In this case, the participants simulated their manipulation of physical
objects when referring to a range of computing constructs. The second metaphor represented ‘computing processes as motion along a path’, in which participants moved their hands along one of three body-based axes when referencing temporal sequences.

As the first study to explore teachers' uses of gesture in computing classrooms, Solomon et al. described teachers' use of embodied representation in the form of gesture, embodied language and tools used in two case studies on teaching recursion (A. Solomon et al. 2020). Using grounded theory, the authors analysed video recordings of undergraduate computing instructors teaching the concept of recursion. They produced a conceptual framework of the gestures that teachers used in computer classrooms, which was the first step toward understanding how gestures and conceptual metaphors supported learners. However, the study could not distinguish between the intentional and non-intentional (i.e., spontaneous) use of embodied representations. Investigating whether the use of embodied representation is intentional might help in critical studies and reflect an understanding of what instructors are trying to communicate. In mathematics, the use of scripted gestures reportedly helps children to retain the knowledge they acquire (Cook, Mitchell, and Goldin-Meadow 2008), which may also apply in computing education. However, researchers in computing education must first understand and explore the types of gestures that teachers and students produce spontaneously in the classroom and they should also determine the effects of these gestures on learning.

2.9 CONCLUSION

This chapter has provided an overview of CS education in the UK at primary school level from the perspective of both students and teachers and the programming environments used in those schools. The chapter also demonstrates the potential benefits of tangible manipulation including TUIs and the use of gestures to support learning in computer science for young children. A review of existing programming environments and previous empirical work that has evaluated TUIs and GUIs for children showed mixed results regarding the benefits of TUIs. Schools have used GUIs such as Scratch to teach primary school student programming because of their availability. However, teachers believed in
the usefulness of physicality in CS even though the evidence for its benefit is mixed and TUIs are considered costly, hard to deploy and to maintain for schools. Therefore, the first overall aim of this thesis is to further examine the effects of physicality in computing education for primary school children in terms of learning outcomes, attitude, enjoyment. The review also indicates, both empirically and theoretically, that there are many potential benefits associated with the use of gestures in a variety of educational situations, especially those involving learning, problem-solving and explaining. These benefits have been much studied in domains such as mathematics, but only partially in computer science, which is a related domain. Therefore, the second overall aim of this thesis is to study the incidence and types of gesture that occur when children learn computer science. The next chapter presents Study 1 which investigated the role of embodied representations, interface type and the use of gestures. Specifically, it examines RQ1, the effect of physicality as an input on learning outcomes (LO), attitude, enjoyment in primary school children aged six-to-seven years by comparing TUIs and GUIs that controlled a physical robot. Additionally, it investigates RQ2, the effect of interface type (TUI or GUI) on the frequency of children’s production of gestures, children’s gesture types, use of conceptual metaphors and the relation between gestures and learning gains.
CHAPTER 3

STUDY 1\textsuperscript{1}: COMPARING TANGIBLE AND GRAPHICAL PROGRAMMING BLOCKS FOR PRIMARY SCHOOL STUDENTS

3.1 INTRODUCTION

Chapter 2 (Section 2.1) described existing research on teaching computing to primary school students and highlighted the potential of using embodied approaches (tangible manipulation and gestures) to support learning of fundamental programming concepts (Section 2.3). This chapter describes Study 1, which investigated the differences between two specific types of TUI and GUI block-based programming environments: a TUI-PR programming environment with physical blocks that a user manipulated by hand and a GUI-PR with virtual blocks that could be manipulated using a touchscreen. In this study, both programming environments were used to control a physical robot. A specific comparison between the TUI and GUI was undertaken to examine further aspects, such as looking at the input and output elements needed. A key motivation is that this understanding will help teachers to choose between readily available GUIs or investing in specialized TUI kits, which are associated with higher prices and maintenance costs to achieve the required learning objectives.

The study addressed both main research questions in this thesis, which are:

- **RQ1:** What are the key differences in how TUI and GUI programming environments support the development of programming skills for students aged six-to-seven years old?

\textsuperscript{1} This chapter contains parts of the text of the following published papers:
RQ1.1: Are there any differences in learning outcomes, attitudes toward computing and enjoyment of computing?
RQ1.2: Are there any differential gender effects in the above measures?

RQ2: How do young children use spontaneous gestures while learning programming and what is the relationship between gesturing and learning outcomes?

RQ2.1: How does interface type affect the use of children’s spontaneous gestures while completing programming tasks?
RQ2.2: How are young children's learning outcomes related to the frequency of spontaneous gestures while completing programming tasks?
RQ2.3: What are the types and purposes of young children's spontaneous gestures while completing programming tasks?
RQ2.4: How are gestures used to represent an abstract programming concept (iteration)?

The findings for RQ1 suggest that the GUI-PR yielded higher learning gains, while the TUI-PR improved attitudes toward computing. No difference was identified between the interfaces in terms of enjoyment outcomes, as published in Paper 1 (Almjally, Howland, and Good 2020a). Additionally, the findings for RQ2 suggest that the interface type had no effect on children's spontaneous gestures and that greater numbers of gestures were associated with greater learning gains compared to lower numbers of gestures. Additionally, this study describes children's spontaneous gesture types, locations and purposes, observing the conceptual metaphors that students invoked when referring to the iteration. The gesture-related analysis was published in Paper 2 (Almjally, Howland, and Good 2020b).

3.2 METHODS

The study used mixed methods and a between-groups research design. For the first research question, the independent variables were (i) interface type (TUI-PR and GUI-PR) and (ii) gender. Gender was considered due to the splitting in the classroom in Saudi Arabia and the inconclusive results regarding the effect of interfaces type on gender (see
section 2.8.1). This produced four groups: TUI-PR girls (TUI-PR-F), TUI-PR boys (TUI-PR-M), GUI-PR girls (GUI-PR-F) and GUI-PR boys (GUI-PR-M). The dependent variables were 1) learning gain, measured by the difference between pre- and post-test scores, 2) attitudinal change, measured by the difference in pre- and post-activity attitudinal survey scores; 3) enjoyment, measured by participants' scores on the post-activity enjoyment survey; 4) number of gestures produced by participants; 5) session time (from the video); 6) interaction time; and 7) number of attempts (from application log data).

For RQ2.1, the independent variable was (i) interface type and for RQ2.3 it was (i) gesture frequency. Participants completed a pre-test and attitudinal survey (a 45-minute learning activity, which included programming a robot), a post-test, and attitudinal and enjoyment surveys. The learning session was video-recorded, and each participant's spontaneous gestures were recorded and coded. Participants were not asked or encouraged to use any gestures – their use was spontaneous and unprompted. When children used their hands or body to debug their code and/or tried to comprehend programming tasks or explain the task, their gestures were recorded and coded. Given that previous research suggested that teachers' use of gestures can benefit student learning (Church, Ayman-Nolley, and Mahootian 2004; Valenzeno, Alibali, and Klatzky 2003), researcher gestures – if any – were recorded and coded to check for any difference in gesture frequency between the groups. The dependent variables were (i) number of gestures and (ii) learning gain.

Participants' mean school mathematics and science scores were added as a participant variable and used to implement matched random assignment to groups as described in the next section.

3.2.1 Participants and Setting

The study was conducted in a Saudi Arabian primary school during normal hours. A pre-test and pre-survey were completed in participants' classrooms. The learning activities, post-test, post-survey, and enjoyment survey were completed in a quiet room in the school. Two classes – both single gender – were invited to take part, with a total of 44 students
(22 females and 22 males) aged six-to-seven. Of these students, 42 students' parents gave informed consent for their children to participate in the study. The 42 participants were paired with same-gender students based on their similar average scores in school mathematics and science (selected because they are related to computational thinking skills (Wing 2010)). There are no standardised tests for students of this age in Saudi Arabia, so students' average coursework scores in these subjects were used. Participants were ranked based on their mean scores and assigned to conditions using blocked randomisation. In one case a request was made by a parent for two friends to be paired together, which was granted. The groups were manually adjusted to account for this.

Table 3.1 shows the participant distribution across conditions, including the mathematics and science mean scores (M&S M) and standard deviations (M&S SD).

<table>
<thead>
<tr>
<th></th>
<th>GUI-F</th>
<th>GUI-M</th>
<th>TUI-PR- F</th>
<th>TUI-PR -M</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>N</strong></td>
<td>12</td>
<td>12</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td><strong>Pairs</strong></td>
<td>6</td>
<td>6</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td><strong>M&amp;S M</strong></td>
<td>81.8%</td>
<td>84.6%</td>
<td>81.6%</td>
<td>83.1 %</td>
</tr>
<tr>
<td><strong>M&amp;S SD</strong></td>
<td>6.28</td>
<td>7.21</td>
<td>6.61</td>
<td>6.30</td>
</tr>
</tbody>
</table>

### 3.2.2 Materials

An activity sheet was used in the learning activity with a Samsung Galaxy Tab E tablet device for managing the two interfaces (TUI-PR and GUI-PR). For the TUI-PR group, the experimental setup also consisted of a physical robot, a mat, programming tiles, physical programming blocks and two TUI-PR components: a mirror to reflect the area in front of the tablet into the front facing camera and then uses computer vision to recognise pieces placed in this area. The mirror fits over the tablet's camera (similar to the Osmo system: https://www.playosmo.com) and a tablet holder (see Figure 3.1).
The GUI-PR consisted of an Android application with a graphical programming block and a graphical robot. We used pre-and post-attainment tests and attitudinal surveys, a post-activity enjoyment survey and a GoPro video camera for data collection.

**SYSTEM DESCRIPTION**

The system was based on an existing programming environment, Tica (Wilkie and Good 2017). It consisted of TUI-PR programming blocks created using a 3D printer, GUI blocks on the tablet’s touchscreen, a physical robot that can be programmed using either language, a mat on which the robot moves and various tiles (e.g., ‘start’, ‘finish’) that can be placed on a grid on the robot’s mat to create tasks for it to complete. The two versions of the blocks were designed to be similar (see Figure 3.1). There were two types of blocks: action blocks and repeat blocks. Three action blocks were used to move the robot (e.g., forward (move one square), followed by turn (right or left); a noise block that could produce a buzzing sound and a repeat block which repeated the action blocks placed inside it twice.

The application offered a GUI programming screen, which was also used to communicate with the robot. The app also recorded task data, including the number of activities completed per participant, activity completion time, number of attempts made per activity and the commands used in each attempt. The GUI had a construction area at the top of the screen and a block list on the bottom left. The user selected a block by dragging it to the construction area. In the TUI-PR, a block could be selected by physically placing it in front of the tablet. The play button ran the code on each interface and a dialogue box
appeared to update the robot (e.g., 'robot is running’). When the robot stopped and if the goal was achieved, the system went to the 'task complete' window; if the goal was not achieved, the user was alerted. The mBot robot was used because it is an open-source educational robot based on Arduino, which was also suitable for the age group (Makeblock 2018). An orange arrow was added to the top of the physical and graphical robot used on the activity sheet to indicate the direction the robot faces (Figure 3.2). The robot moved on the mat or screen grid (a 5×5 grid), which had four types of tiles to configure the tasks: 1) 'S' (the starting point); 2) flag (indicate the end/goal); 3) a speaker (when the robot reaches this tile, it must stop and produce a noise); and 4) 'X' (blocks the robot/the robot cannot go to 'X' square) and 5) grey tiles used to make a specific path for the robot to follow (see Figure 3.2 and Figure 3.3).

![Figure 3.2 mBot with orange arrow and activity sheet (left) and tile types (right)](image)

Figure 3.2 mBot with orange arrow and activity sheet (left) and tile types (right)
Figure 3.3 mBot robot and robot mat

Problem-Solving Approach/Log Data

The application recorded each pair's activity log, including their activity completion time, number of attempts made per activity, programming blocks used in each attempt and attempt time. Attempt time referred to the time a participant took to decide on a solution and put together the blocks before pressing the play button. Using log data, the interaction time can be calculated. This refers to the total time that pairs spent across all attempts interacting with the programming blocks (i.e., from the moment they started working on an attempt until they clicked the play button to test their code). This measure provides an indication of how participants interacted with the interfaces and it may be used to infer details regarding their programming approach. In our case, programming blocks in an environment the amount of 'tinkering' that occurs, where 'tinkering' is a term in pedagogical theory that suggests learners need opportunities to experiment and explore ideas by tinkering with the artifact leading learners to self-directed learning rather than having a pre-established plan.
LEARNING OUTCOMES (LO)

As there is no standard primary computing curriculum in Saudi Arabia (Ministry of Education; Tatweer Company for Educational Services 2018), the learning outcomes were based on the English National Curriculum for Computing (specifically, the Key Stage 1 curriculum), since England is one of the few countries that teaches computing in primary school as standard. The following learning outcomes were defined, targeting program creation, debugging, and comprehension.

- **Create and debug program:**
  - **Describe:** Articulate program goals.
  - **Simple coding:** Write a simple program by selecting and placing the right action commands (one or more actions) and placing them correctly.
  - **Complex coding:** Write an advanced program by selecting and placing the right sequence of action commands in one iteration (including repeat and action commands).
  - **Debug:** Anticipate what will happen, find out precisely what does happen and pinpoint and correct bugs (Carver and Klahr 1986).

- **Comprehension:** Explain a simple program.

ATTAINMENT TEST

Two semantically identical versions of the attainment test were developed, which varied only in terms of the structure of the task maps (see Appendix A). In addition, both focused on testing LO. Approximately one-half of the participants received *version A* as the pre-test and *version B* as the post-test, whereas the remainder received the tests in reverse order. This was an extra measure to ensure that there were no unintended differences between the pre and post-tests in terms of difficulty. All the images in the tests (i.e., grids and blocks) were similar to the learning activity images and the tests were worth 27 points in total. Each test contained two multiple-choice questions with four potential answers and four open-ended questions. The questions (in both versions) were as follows:
Q1. Look at the picture below (Figure 3.4). The arrow shows where the robot is and which way it is facing: [3 points]

Figure 3.4 Question 1

A. What shape will the robot do to get to the Finish square? [LO: Familiarizing participants with the questions and grid]

B. How many moves to the Right will the robot have to do to go to the Finish square? [LO: Describe]

C. What are the first 2 moves the robot will have to do to go to the Finish square? [LO: Simple coding]

Q2. Which of the programs below will get the robot to make noise and go to the Finish square? [LO: Trace/ Debug]
Q3. Which square of the grid will the robot go to, when following the program below?  

[LO: Trace/ Debug] Use the given sticker to show which will be the Finish square (see finger).
Q4. Write a program to make the robot go to the Finish square. (Use the given stickers see Figure 3.9) [LO: Simple coding]

You can use as many stickers as you need
Q5. Now make the robot go to the Finish square using 5 blocks only. (Use the given stickers). [LO: Complex coding]
Q 6. Which solution is more advanced/ better and why? [LO: Comprehension]

Solution 1

Solution 2

Why ………………………

Figure 3.9 Stickers used in attainment
ATTITUDINAL SURVEY

For the attitudinal survey (see Appendix B), five statements were selected based on their relevance to computing with children, taken from a survey used to assess a game-making project's effect on attitudes to computing (Robertson 2013). These statements were 1) 'I like computing', 2) 'I am good at computing', 3) 'I like the challenge of computing', 4) 'Computing is fun' and 5) 'I want to find out more about computing'. Answers were recorded on a 5-point Likert scale, presented as a series of faces ranging from sad to happy.

ENJOYMENT SURVEY

The enjoyment survey (see Appendix B) used in (Yannier 2016) contains three questions. Question 1 assessed children's liking of the interface using a 5-point Likert scale ('I didn't like it at all', 'I didn't like it', 'It was okay', 'I liked it' and 'I liked it very much'). Question 2 focused on whether the child wanted to try the activity again, while Question 3 asked if the child will tell their friend about the system. Both questions had three choices: 'No', 'Maybe' and 'Yes'. For assistance, the scales used emojis to symbolise each reaction to help the participants better understand the choices on the scale.

3.2.3 Learning Activities

Six activities with different complexity levels were used in the study, as shown in Figure 3.10. Each activity aimed to fulfil the learning outcome by familiarising the student with or developing programming concepts (e.g., sequences, comprehension, debugging or iteration). For example, Activities 1-3 focused on creating and debugging simple programs, while Activities 4-6 focused on creating and debugging complex programs. All the activities satisfied the aim of describing debugging and comprehension.

Activity complexity was estimated based on the intrinsic cognitive demands of the activity. The definition of cognitive activity is that activity requires a person to mentally process new information (i.e., acquire and organise knowledge/learn) and allows them to recall, retrieve that information from memory and to use that information at a later time in the
same or similar situation (i.e., transfer). Consequently, Activity 3 was categorised by the researcher as a high complexity and Activity 6 as moderate complexity activity.

Activities on the activity sheet and the robot mat using the tiles (shown in Figure 3.2 and Figure 3.2 for TUI-PR users or graphical grid for GUI-PR users. In Activity 1, the robot must reach a destination square by moving forward four times. The aim of the learning activity is to introduce participants to the system, to the activity and to the activity sheet, allowing them to be comfortable with the setting. It also allowed participants to gain confidence by solving a simple programming task (in this case, placing a block four times). In Activity 2, students need to use the turn and noise blocks to complete the task. In Activity 3, students must use turns and rotations to find the shortest path to the flag and avoid the 'X' tiles (e.g., Figure 3.2). In Activity 4, the robot must make a noise twice using the repeat command. Activity 5 requires the use of turns and the repeat command. Activity 6, the robot has to follow the grey squares to reach the flag, requiring a more complex program that included turn blocks, some of which needed to be placed within a repeat command.
### Activity 1

![Activity 1 Image]

4 Forward

### Activity 2

![Activity 2 Image]

Right, Forward, Noise, Left, Forward

### Activity 3 (High Complexity)

![Activity 3 Image]

Right, Right, Forward, Right, Forward, Noise, Forward

### Activity 4

![Activity 4 Image]

Repeat {Forward, Noise}, Forward

### Activity 5

![Activity 5 Image]

Repeat {Left, Forward}

### Activity 6 (Moderate Complexity)

![Activity 6 Image]

Repeat {Forward, Right}, Forward

---

#### 3.2.4 Procedure

The study ran from November to December of 2018. Participants completed two sessions. In the first session, lasting 30 minutes, students first received an introduction and then the pre-survey was distributed, completed and collected. The pre-test and stickers required for answering Questions 3–5, were then distributed to participants (with one class receiving version A and one class receiving version B). The researcher read each question and multiple-choice options aloud to the whole class, as students were not fluent in reading, but students completed their test sheets individually at their desks. No time limit was given for the tests. The researcher made sure that everyone had finished each question before moving on to the next question.
The second session took place in a quiet room in the school and lasted approximately 45 minutes. In this session, pairs of participants completed the learning activities, followed by a post-test, a post-attitudinal survey and an enjoyment survey, which they completed individually. The procedure for both interfaces was the same: pairs of participants were first given an overview of the session and the programming environment, an explanation of the functionality of the action blocks and were asked to solve Activities 1–3. For Activity 4, involving iteration, participants first completed it without using the repeat block. Then, the researcher introduced the repeat block and explained why iteration can be useful. She asked students to solve Activity 4 again using the repeat block. For the two other activities involving iteration (Activities 5 and 6), participants were asked to solve the activity twice: once without the repeat and once with the repeat.

Before beginning the programming, participants were asked to explain orally what they wanted the robot to do, discuss their solution in pairs and work together to program their solution. The aim of asking participants to explain their solution was to capture and analyse their use of spontaneous gestures. Nevertheless, the researcher did not ask the participants to gesture in any part of the session.

When disagreement occurred within pairs, the researcher asked one participant to clarify their choice of blocks and then the other participant was asked to do the same. At this point, one participant would usually become convinced of their partner's explanation and they would continue programming. If not, they were asked to explain why they were not convinced and to suggest another solution. There were only a few cases where intervention was necessary.

When participants requested help, they were assisted. The researcher explained the issue and motivated them to think about the solution. No limit was applied to the number of attempts the participants could make. The researcher cheered and offered encouraging feedback each time an activity was completed. Participants mostly showed excitement about their accomplishments such as dancing when the solution is correct.
After completing the learning activities as pairs, the participants individually completed the enjoyment survey, the post-attitudinal survey and finally the post-test. The questions were read by the researcher, as in the procedure for the first session.

### 3.3 Analysis

#### 3.3.1 Test Schema

To analyse students' answers to the open-ended questions (Q3–Q6), a coding scheme was developed. The code was read from top to bottom, as was demonstrated in the programming environment. A total score based on placement, block selection and difficulty level were assigned. Incorrect block scores received no points, while one point was allocated for the correct placement of an action block. The repeat blocks were worth more points because they relate to an advanced programming concept. In Question 6, the participants were asked to evaluate their two solutions to Questions 4 and 5 by selecting the better solution and writing their reasons.

**Test's Coding Reliability**

The first author coded the test data and then a random sample of 20% of the answers to Questions 4 and 5 was generated for second coding conducted by another PhD student. Inter-rater reliability was determined using *Cohen's Kappa*. The Kappa value was .73, indicating good agreement (Fleiss, Levin, and Paik 2004).

#### 3.3.2 Overview of Gesture Coding Rubrics

Unfortunately, of the 21 sessions, the recordings for four sessions were damaged, so only 17 sessions were coded (with a total of 34 participants). Table 3.2 shows participant distribution across conditions.

<table>
<thead>
<tr>
<th></th>
<th>GUI-PR-F</th>
<th>GUI-PR-M</th>
<th>TUI-PR- F</th>
<th>TUI-PR -M</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>10</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Pairs</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>M&amp;S M</td>
<td>84.5 %</td>
<td>84.3%</td>
<td>82.5 %</td>
<td>82.5 %</td>
</tr>
<tr>
<td>M&amp;S SD</td>
<td>4.4</td>
<td>6.2</td>
<td>6.5</td>
<td>4.6</td>
</tr>
</tbody>
</table>
17 videos of learning activities were coded with a mean duration of 31 minutes. The sessions included three actors: the researcher and two participants.

The children were novices and lacked any background in CS. Therefore, the researcher’s strategy was to explain programming concepts using the same pedagogical approach used in schools, and to prioritize a natural delivery of the content. For this reason, no prior plan was made of how and which gestures the researcher would use. Instead, a spontaneous position was adopted. However, the literature has suggested that teachers’ gestures can benefit student learning. Therefore, researcher was considered an actor, and their gestures were recognised and quantified. For example, when teachers use gestures in mathematics, they help students make connections between representations such as equations (Yeo et al. 2017). For more about the impact of teacher gesturing, see section 2.6.4.

The videos were coded using ELAN video annotation software (Wittenburg et al. 2006). A unit of a gesture was defined as the duration from the movement's start until the hand returned to its resting position (McNeill 1994). We excluded participants' self-adapting motions from the analysis (e.g., head-scratching or changing hand position from lap to desk), as they lacked semantic attachment.

Gestures were coded based on their type, location and purpose. First, we created a coding scheme to identify the gesture types based on the mathematic gesture taxonomy developed by (Martha W. Alibali and Nathan 2012), shown in Table 3.3. The locations where the gestures were performed are shown in Table 3.4 and the apparent purpose of the gesture was coded using the scheme shown in Table 3.5.

<table>
<thead>
<tr>
<th>Gesture types</th>
<th>Code</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pointing (Deictic)</td>
<td>Pointing-</td>
<td>Deictic gesture, reflecting grounded cognition in the physical environment e.g., pointing to the code, or the robot path.</td>
</tr>
<tr>
<td></td>
<td>gesture</td>
<td></td>
</tr>
</tbody>
</table>
Use robot to physically demonstrate its movement.

Representational gesturing* with hands e.g. using hands to indicate turn (includes one instance using a foot).

Representational gesturing* with body e.g. rotating the whole body to indicate turn.

* Representational gesturing: A gesture handshape or motion trajectory, depicting aspects of their meaning either literally or metaphorically either by hand or body (Martha W. Alibali and Nathan 2012)

Table 3.4 Gesture location

<table>
<thead>
<tr>
<th>Location</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity sheet (AS)</td>
<td>An A4 paper, which shows the task that the participants need to solve (see Figure 3.2).</td>
</tr>
<tr>
<td>Robot mat (RM)</td>
<td>An A0 mat on which the robot moves (see Figure 3.3).</td>
</tr>
<tr>
<td>Code</td>
<td>The programming blocks participants use to solve the task</td>
</tr>
<tr>
<td>Air</td>
<td>Gestures performed in the air, without reference to the AS or RM.</td>
</tr>
</tbody>
</table>

Table 3.5 Gesture purpose

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direction</td>
<td>Demonstrate the direction of robot movement such as forward, backward, left turn, right turn, or double turns</td>
</tr>
<tr>
<td>Indicate object</td>
<td>Indicate the tiles in programming task such as a square, flag, noise, robot</td>
</tr>
<tr>
<td>Simulation</td>
<td>Simulate the route the robot will follow to complete the task</td>
</tr>
<tr>
<td>Counting</td>
<td>Counting squares</td>
</tr>
<tr>
<td>Iteration</td>
<td>Represent an iteration</td>
</tr>
</tbody>
</table>
GESTURE CODING RELIABILITY

After the author had coded all the video data, a random sample of 20% was generated for second coding by another PhD student. Inter-rater reliability was determined using Cohen's Kappa scores, with a strength agreement (Fleiss, Levin, and Paik 2004). The scores were as follows: for Type = .724, Good; for Location = .897, Excellent; and for Purpose = .742, Good.

3.4 RESULTS

This section presented the results that contributes to RQ1 (Section 3.4.1 and Section 3.4.2) and RQ2 (Section 3.4.3).

3.4.1 Effect of Interface Type and Gender on Learning Outcomes, Attitudes and Enjoyment

This section presents the results of the attainment tests and surveys to answer RQ1: When comparing TUI-PRs and GUI-PRs for programming activities with children aged six-to-seven, how does interface type influence children's learning outcomes, attitudes towards computing and enjoyment of programming activities? what are the differential gender effects in the above measures?

Due to the small sample size (n=42) and the nature of children's different developmental levels, some of the data did not meet all independent t-test assumptions and failed to meet normality assumptions. Therefore, we assessed the degree on non-normality in the data using the Fisher skewness coefficient and Fisher Ketosis coefficient (z values). Most of the data had an acceptable range of normality, with z values in the range of +1.96, which for a small sample size (n<50) is considered not significantly different from the normal distribution (Kim 2013; Pett 2016). In these cases, we used parametric tests. Summary charts showing results for the tests and surveys by condition can be seen in Figure 3.11.
Effect of interface on learning outcomes

To measure learning outcomes, we calculated the normalised learning gains by considering the maximum possible gain or loss given the pre-test score (Marx and Cummings 2007), using the following formula: \( 100 \times \frac{\text{post} - \text{pre}}{100 - \text{pre}} \). Before analysing the results, one outlier was identified and dealt with by adjusting the value to be closer to the rest of the data set whilst maintaining rank order, in line with the Winsorisation approach (Ghosh and Vogt 2012; Kwak and Kim 2017). An independent-samples t-test was run to determine differences in normalised learning gains between interface types. The GUI-PR users had greater normalised learning gains (\( M=9.5, \text{SD}=4.7 \)) than the TUI-PR users (\( M=6.5, \text{SD}=4.5 \)), a statistically significant difference of 0.293 (95% CI, -0.0002 to 0.058), \( t(40) = 2.036, p = .048 \).

Effect of gender on learning outcomes

To determine if gender differences exist in the pre- and post-test scores, an independent-samples t-test was run. On the pre-test, males achieved higher scores than females (\( M = 4.51, \text{SD} = 3.05 \), \( M = 3.61, \text{SD} = 1.98 \)) respectively, but the difference was not statistically significant (\( M = .9037, 95\% \text{ CI} [-.73, 2.54], T(32.1) = 1.12, p = .269 \)). On the post-test, males (\( M=12.15, \text{SD}=3.70 \)) achieved higher scores than females (\( M=11.92, \text{SD}=4.50 \)).
SD=4.87), but the difference was not statistically significant (M =.231, 95% CI [-2.48, 2.95], t (40) =.172, p =. 864).

Although the males’ scores in the pre- and post-tests were higher than females’, the normalised learning gains for females were higher than males (M = 8.53, SD =4.9), (M = 7.91, SD = 4.72); however, the difference was not statistically significant (M =.621, 95% CI [-2.38, 3.62], t (40) =.414, p =.679).

**EFFECT OF GENDER AND INTERFACES TYPE ON LEARNING OUTCOMES**

The average normalised learning gains for the groups by gender were (M, TUI-PR) (M=5.34 ,SD = 2.48), (F, TUI-PR) (M=7.5, SD=5.5), (M, GUI-PR) (M=9.62 ,SD=5.16) and (F, GUI-PR) (M=9.4 ,SD=4.40). A two-way ANOVA was run to examine the effects of interface and gender. The interaction effect between interface and gender on normalised learning gains was not statistically significant, F (1, 38) = .678, p = .415, partial η² = .018.

**EFFECT OF INTERFACE ON ATTITUDINAL CHANGE**

There were two outliers on attitudinal change grouped by condition and four outliers grouped by gender; these were adjusted using the approach described in the previous section. Analysis of the attitudinal survey responses was conducted on the sum of the five questions, creating a single attitudinal measure on a 25-point maximum and 5-point minimum scale. An independent-samples t-test was run to determine if the interface type affected children's change in attitude toward computing. The TUI-PR group showed greater increase on scores (M = 3.169, SD = 3.22) than the GUI-PR group (M =.958, SD = 3.02), with a statistically significant difference in mean attitudinal improvement scores between the two interfaces (M=2.208, 95% CI, [-.247, 4.16], t (40) = 2.27, p =.028).

**EFFECT OF GENDER ON ATTITUDINAL CHANGE**

An independent-samples t-test was run to determine differences in attitudinal-improvement scores between males and females. The experiment had more impact on the females’ attitude (M=2.409, SD=2.9) than males’ (M=.1.35, SD=3.58), but showed no statistically significant difference, M=1.059, 95% CI [-.978, 3.096], t (40) = 1.05, p =.300).
EFFECT OF INTERFACES TYPE AND GENDER ON ATTITUINAL CHANGE

The average attitudinal improvement for the groups by gender was (M, TUI-PR) (M= 3.25, SD = 3.89), (F, TUI-PR) (M= 3.1, SD=2.8), (M, GUI-PR) (M=.08, SD=2.88) and (F, GUI-PR) (M= 1.83, SD=3.04). A two-way ANOVA was conducted to examine the effects of interface and gender. The interaction effect between interface and gender on attitudinal improvement scores was not statistically significant, F (1, 38) = .949, p =.336, partial η2 = .024.

EFFECT OF INTERFACE ON ENJOYMENT

As the data were not normally distributed by interface type, or by gender, a Mann-Whitney U test was used to determine if there was a difference in enjoyment scores across conditions. The GUI-PR (mean rank= 21.9) and TUI-PR (mean rank= 20.8) were not significantly different, U = 204, z = -327, p =.744. An additional Mann-Whitney U test was run to examine differences in enjoyment scores between genders. The mean rank was not significantly different (female mean rank = 23.3, male mean rank= 19.45), U = 179, z = -10154, p =.249.

3.4.2 Overview of Time Spent on Session, Interaction and Attempts

Before answering RQ2, an analysis of the log data and session times was conducted to find out if there were any differences between the groups besides the interface types. This section presents an overview of the quantitative data collected from the application for 21 groups (interaction time, number of attempts) and video recording (session time). For the session time, four recording were missing due to technical error with the camera. Thus, 17 groups were recorded and reported for session time and gestural analysis. For the descriptive analysis of overall time spent on session, interaction and number of attempts for condition, see Table 3.6 and for breakdown by condition and gender, see Table 3.7.

Table 3.6 Descriptive analysis of times and attempts by condition (N is the number of pairs)

<table>
<thead>
<tr>
<th></th>
<th>TUI-PR</th>
<th></th>
<th>GUI-PR</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>M</td>
<td>SD</td>
<td>N</td>
<td>M</td>
</tr>
</tbody>
</table>
Table 3.7 Descriptive analysis of times and attempts by condition and gender (N is the number of pairs)

<table>
<thead>
<tr>
<th></th>
<th>TUI-PR</th>
<th></th>
<th>GUI-PR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TUI-PR-F</td>
<td>TUI-PR-M</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N</td>
<td>M</td>
<td>SD</td>
</tr>
<tr>
<td>Session time</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>42.3 mins</td>
<td>6.2</td>
</tr>
<tr>
<td>Interaction time</td>
<td>5</td>
<td>16.4 mins</td>
<td>5.7</td>
</tr>
<tr>
<td>Number of Attempts</td>
<td>5</td>
<td>19.4</td>
<td>4.3</td>
</tr>
</tbody>
</table>

EFFECT OF AN INTERFACE ON SESSION TIME

An independent samples t-test was run to check for any significant differences in session times between interface types after the test assumption was satisfied. The difference was not statistically significant ((95% CI, -3.9 to 13.02), t(15) = 113, p = .275).

The average interaction time for the groups by gender were (M, TUI-PR) M=36.25 (SD = 14.3), (F, TUI-PR) M=42.35 (SD=6.2), (M, GUI-PR) M=34.0 (SD=3.1) and (F, GUI-PR) M=35.4 (SD=6.5). A two-way ANOVA was run to examine the effects of interface and gender. The interaction effect between interface and gender on session time was not statistically significant, F (1,13) =.324, p =.579, partial η2 = 23.25.

EFFECT OF AN INTERFACE ON INTERACTION TIME

Before analysing the interaction time, three outliers were found in the GUI-PR groups and dealt as we did in the previous sections (3.4.1.1). No significant difference was observed between the two groups' interaction times using independent samples t-test (95% CI, -2.9 to 3.5), t (19) = 186, p = .854).

The average interaction time for the groups by gender were (M, TUI-PR) M= 14.2 (SD =2.5), (F, TUI-PR) M= 16.4 (SD=5.7), (M, GUI-PR) M= 13.7 (SD=1.4) and (F, GUI-PR) M=16.7 (SD=1.5). A two-way ANOVA was run to examine the effects of interface and
The interaction effect between interface and gender on interaction time was not statistically significant, F (1,17) = .066, p = .800, partial η² = .763.

EFFECT OF AN INTERFACE ON ATTEMPTS

One outlier was found in the TUI-PR group, it was adjusted as we did in the previous sections (3.4.1.1). After the independent samples t-test assumption was satisfied, a test was run to determine whether there is a difference between the number of attempts by interface type. No statistically significant difference was found (95% CI, -5.8 to 3.92), t (19) = -1.832, p = .83).

The average attempt for the groups by gender were (M, TUI-PR) M=18.50 (SD =1.2), (F, TUI-PR) M= 19. 4(SD=4.), (M, GUI-PR) M= 20.17(SD=2.6) and (F, GUI-PR) M=23.3 (SD=3.8). A two-way ANOVA was run to examine the effects of interface and gender. The interaction effect between interface and gender on number of attempts gains was not statistically significant, F (1,17) = .592, p = .452, partial η² = 6.55.

3.4.3 Use of Spontaneous Gestures While Solving Programming Tasks

This section presents the result of the gestural analysis of the 17 video sessions in order to answer RQ2: ‘How do young children use spontaneous gestures while learning programming, and what is the relationship between gesturing and learning outcomes?’

RQ2.1 HOW DOES INTERFACE TYPE AFFECT CHILDREN'S USE OF SPONTANEOUS GESTURES WHILE COMPLETING PROGRAMMING TASKS?

An independent samples t-test was conducted to determine whether interface type affected children's spontaneous gestures. The mean number of gestures produced by the GUI-PR group (M=28.56, SD=12.55) was lower than that produced by the TUI-PR group (M=32.13, SD=16.41), but this difference was not statistically significant. An independent samples t-test was also run to determine whether there was a difference in the frequency of researcher gestures across the interface types. The mean number of gestures produced by the researcher was largely the same across the GUI-PR group (M=43.67, SD=15.88) and the TUI-PR group (M=43.38, SD=15.08) and the difference was not statistically significant.
RQ2.2 How are young children’s learning outcomes related to the frequency of spontaneous gestures while completing programming tasks?

Given the positive benefits of spontaneous gestures for learning reported in the literature, we were interested, first, in whether there were participants in this dataset who could be characterised as either ‘high frequency gesturers’ or ‘low frequency gesturers’ and second, in whether there was a relationship between gesture frequency and learning gain. We therefore divided the dataset by quartile according to the frequency of gesture, characterising participants in the upper quartile as high frequency gesturers and those in the lower quartile as low gesturers. An independent samples t-test was conducted to determine if there were individual differences in the computing knowledge between the groups the low and high gesturers in the pre-test. The mean of the pre-test scores in the low frequency gesturers group (M=4.48, SD=2.56) was slightly higher than in the high gesturers group (M=4.07, SD=2.56), but this difference was not statistically significant.

Normalised learning gains were calculated by taking into the account the maximum possible gain or loss given the pre-test score (Marx and Cummings 2007), using the following formula: $100 \times \frac{\text{post-pre}}{100-\text{pre}}$. The mean number of gestures for high frequency gesturers was (M=50.25, SD=10.32), while for low frequency gesturers, it was (M=13.75, SD=2.91). The mean normalised learning gain for high frequency gesturers was (M=11.38, SD=4.96) and for low frequency gesturers it was (M=5.64, SD=3.6). An independent samples t-test showed a statistically significant difference in learning gains between the high-frequency gesturing and low-frequency gesturing groups (t (14) =2.68, p=.019). A further independent samples t-test was run to determine whether there was a difference between the researcher gestures in each frequency group. No statistically significant difference was found. Researcher gestures were considered because the teacher’s gestures might influence learners (Church, Ayman-Nolley, and Mahootian 2004; Valenzeno, Alibali, and Klatzky 2003).

Additionally, a Pearson's product-moment correlation was run to assess the relationship between student’s achievement and number of gestures. There was a no statistically
significant correlation between the school math and science scores which were used to match the pairs and number of gesture, \( r(32) = .037, p = .834 \).

A Pearson's product-moment correlation was run to assess the relationship between students' performance in math and science and in computing. There was no statistically significant correlation between the school math and science scores and learning gains, \( r(32) = .281, p = .108 \).

RQ2.3 What are the types and purposes of young children's spontaneous gestures while completing programming tasks?

We coded 1,019 participant gestures. The gesture type used most frequently by both participants (P) and the researcher (R) was pointing, followed by hand movement, body movement and using the robot to point (see Table 3.8). Gestures were performed most frequently on/near the activity sheet followed by near/above the robot mat, then the code and finally in the air (see Table 3.9). The purposes of the gestures were simulation (42.7%), direction (37.8%), indicate object (11%), counting (6.8%) and iteration (1.6%).

<table>
<thead>
<tr>
<th>Types</th>
<th>Participants</th>
<th>Researcher</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pointing</td>
<td>695</td>
<td>592</td>
</tr>
<tr>
<td>Hand movement</td>
<td>258</td>
<td>100</td>
</tr>
<tr>
<td>Body movement</td>
<td>60</td>
<td>27</td>
</tr>
<tr>
<td>The robot</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Total</td>
<td>1,019</td>
<td>726</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Location</th>
<th>Participants</th>
<th>Researcher</th>
</tr>
</thead>
<tbody>
<tr>
<td>The activity sheet</td>
<td>634</td>
<td>334</td>
</tr>
<tr>
<td>The robot mat</td>
<td>296</td>
<td>197</td>
</tr>
<tr>
<td>The code</td>
<td>56</td>
<td>173</td>
</tr>
<tr>
<td>The air</td>
<td>33</td>
<td>22</td>
</tr>
<tr>
<td>Total</td>
<td>1,019</td>
<td>726</td>
</tr>
</tbody>
</table>
GESTURE TYPE RELATIONSHIP TO GESTURE PURPOSE AND IDENTIFIABLE PATTERNS

To identify patterns that emerged according to participants' gesture types and purpose, cross-tabulation is used (Table 3.10) to show the data for a given gesture's purpose and the most frequent type of gesture used by percentage.

Table 3.10 Cross-tabulation for gestures type and purpose

<table>
<thead>
<tr>
<th></th>
<th>Pointing (Diectic)</th>
<th>Representational</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pointing</td>
<td>Robot</td>
</tr>
<tr>
<td>Simulation</td>
<td>52.5%</td>
<td>66.7%</td>
</tr>
<tr>
<td>Direction</td>
<td>20.9%</td>
<td>33.3%</td>
</tr>
<tr>
<td>Indicate object</td>
<td>16.1%</td>
<td>0%</td>
</tr>
<tr>
<td>Counting</td>
<td>9.9%</td>
<td>0%</td>
</tr>
<tr>
<td>Iteration</td>
<td>.6%</td>
<td>0%</td>
</tr>
<tr>
<td>Total</td>
<td>695</td>
<td>6</td>
</tr>
</tbody>
</table>

Most of the purposes analysed demonstrated a representation of a concrete concept such as simulation, direction, object and counting. However, it is interesting to examine the use of representational gesture (RG)-metaphorical which is the gesture that represents abstract concepts (Martha W. Alibali and Nathan 2012). RG-metaphorical may provide additional information about the speaker’s conceptualisations by externalising dynamic visual imagery to support cognitive activity. For further details on the benefits of gestures, see Section 2.5. Abstract concepts are very common in computing and observing young children representing them spontaneously through gesture is noteworthy. Therefore, the next subsection discusses the gestures that were used to represent the iteration concept.

RQ2.4 REPRESENTATIONAL GESTURE: INSIGHTS ON THE REPRESENTATION OF AN ABSTRACT CONCEPT (ITERATION)

During eight sessions where 1,019 gestures were observed among 34 participants, only 10 participants performed gestures that represented the abstract concept of iteration (12 hand and body gestures in total), as shown in Table 3.11. Pointing to the repeat block was not included in this analysis as we were specifically interested in investigating the use of representational gestures for abstract concepts. The description column categorises the
representational gestures exhibited by each participant while solving the programming task. No significant differences in learning gains were found between the participants who used gestures that represented an abstract concept and the others.

Table 3.11 Iteration representational gesture description

<table>
<thead>
<tr>
<th>Gesture</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hand</td>
<td>Performing a circular motion</td>
<td>Figure 3.12 P1</td>
</tr>
<tr>
<td>Hand</td>
<td></td>
<td>Figure 3.13 P2</td>
</tr>
<tr>
<td>Hand</td>
<td>Repetitive circular movement as if he is stirring something</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>----------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Figure 3.14 P3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hand</th>
<th>Both hands, in a rolling movement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Figure 3.15 P4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hand</th>
<th>Repeated the action command inside the repeat, which is forward, using tilted hand, twice.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Figure 3.16 P4</td>
</tr>
<tr>
<td>Hand</td>
<td>Circular gesture that indicates a container</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------</td>
</tr>
<tr>
<td>Body</td>
<td>Opening the arm in a circular shape and taking one step forward</td>
</tr>
</tbody>
</table>

Figure 3.17 P5

Figure 3.18 P6

### 3.5 Discussion

The data presented in this chapter was analysed with a view to answering RQ1 and RQ2. Here, each question is addressed in turn and the implications of the findings are discussed.

#### 3.5.1 RQ1 What are the Key Differences in How TUI and GUI Programming Environments Support the Development of Programming Skills for Students Aged 6-7 Years Old?

To explore the key differences in how TUI and GUI support the development of programming skills for young children (RQ1), the question is broken into four sub-questions (RQ1.1 and RQ1.2).

**RQ1.1 Are there any differences in learning outcomes, attitudes toward computing and enjoyment of computing?**

This study compared an isomorphic TUI and GUI in a computer-programming activity with primary-school students aged six-to-seven in Saudi Arabia. The study involved three dependent variables: learning gain, attitude change and enjoyment, which were examined as a function of two independent variables (interface type and gender). There is promising
existing research on the relevance of embodied cognition in other subject areas, with studies showing that physical manipulation can benefit learning English and science (Glenberg, Goldberg, and Zhu 2010; Kontra et al. 2015). However, there is not yet evidence of this in CS and our results show that GUI-PR users had significantly greater learning gains than TUI-PR users. This finding is also contrary to a previous study of the effects of interface type on learning outcomes, which found no significant difference between groups (Strawhacker and Bers 2015).

It is unclear why GUI-PR improved learning outcomes in the current study. One possible explanation is learners’ familiarity with tablet usage. In the previous literature that investigated TUIs and GUI, the GUIs were on computer screen and use a mouse as an input device (Sapounidis, Demetriadis, and Stamelos 2015; Strawhacker, Sullivan, and Bers 2013), which makes it difficult to compare with GUI findings in this research. The GUI with tablet allows children to have a direct manipulation of blocks on screen using their hands, which could be more beneficial compared to a mouse and keyboard. This is a common practice in the field of robots, such as LEGO® MINDSTORMS®.

On attitude towards computing measures, the TUI-PR group had a significantly greater improvement. To understand why GUIs might support greater learning gains and TUIs might lead to attitudinal improvements, further work should be carried out. For example, this could involve more detailed examinations of interactions with both types of interfaces during learning activities. Despite the difference in attitude change, no significant difference was found between interface type and enjoyment of the activities. This may be because both interfaces have the same engaging output, namely, a physical robot. Further research should investigate the role of physicality in the output as well as the programming environment. This could be examined using isomorphic interfaces with graphical programming/a graphical robot compared to tangible interaction/a physical robot.

RQ1.2 ARE THERE ANY DIFFERENTIAL GENDER EFFECTS IN THE MEASURES?

There were no statistically significant differences between gender or genders and interfaces for all three dependent variables, in line with (Horn et al. 2009).
3.5.2 RQ2 How Do Young Children Use Spontaneous Gestures While Learning Programming and What is the Relationship Between Gesturing and Learning Outcomes?

To explore how children used their spontaneous gestures (RQ2), the question is broken into four sub questions (RQ2.1 - RQ2.4).

**RQ2.1 How interface type (TUI or GUI) affects children's spontaneous gestures?**

The results did not show evidence of a significant difference between the two interface groups in terms of gesture frequency. There is no evidence from this study that object manipulation encouraged gesturing, but as both interfaces had the same physical output (a robot), further investigation of the role of physicality of the task and gesture would be warranted.

**RQ2.2 How are children's learning outcomes related to the frequency of spontaneous gestures while completing programming tasks?**

The participants in the upper quartile of gesture frequency were characterised as ‘high gesturers’ and those in the lower quartile as ‘low gesturers. The findings showed a statistically significant difference in the learning gains between the high-frequency gesturing and low-frequency gesturing groups. No relationship was observed between the maths scores used to match the pairs and the number of gestures used, which suggests that the high-performance students in math were not high gesturers.

This is a promising finding which suggests that gestures might have a role in supporting learning in computing (Antle and Wise 2013; Bruner 1966). This is in line with the empirical work that has been carried out in mathematics (Martha W. Alibali and Nathan 2012). Additionally, in computing education, Solomon et al. in 2018 found that students who did not use gestures when they explained control flow could not correctly predict the output (A. Solomon et al. 2018). Although the study did not investigate possible causation, this might mean that encouraging students to be active and use gestures leads to improvement in their reasoning about the answers or simply that greater knowledge and understanding leads to more gesturing.
No relationship was identified between the prior school maths test and learning gains. This may be because the test scores were unreliable as they were not based on standardised tests; rather, they represented the teacher’s subjective evaluation of a child’s performance in the class.

Further work is needed to investigate the causal role of gestures in computing classrooms, particularly the question of whether gestures assist and enable learning, whether gestures are indicative of existing ability or understanding, or whether gestures are simply inconsequential side-effects of learning and information processing. Additionally, future researchers may examine the role of specific gestures instead of focusing only on spontaneous gestures.

**RQ2.3 WHAT ARE THE TYPES AND PURPOSES OF YOUNG CHILDREN’S SPONTANEOUS GESTURES WHILE COMPLETING PROGRAMMING TASKS?**

The data showed that participants used pointing or *deictic* gestures most frequently. The main purpose of deictic gestures is to reflect the grounding of cognition in the physical environment. According to McNeill, these gestures physically link speech and associated mental processing gestures to the physical environment (McNeill 1994). This was found in mathematics and described by (Martha W. Alibali and Nathan 2012). In computing education, Solomon et al. found that pointing gestures help to trace the code (A. Solomon et al. 2018). In this study, students’ use of pointing gestures was particularly high when simulating the robot's actions. Deictic gestures were used to help students clarify the object and its orientation to which they were referring while talking. Finally, pointing was used to count the number of squares that the robot would follow, which is a simple illustration of the grounded cognition theory that tools from the environment are used to off-load cognitive work (M. Wilson 2002). The second type of gesture used was representational gestures using hand or body to represent an abstract concept. This type of gesture was used to demonstrate direction, such as a turn or forward action, or to simulate the robot's movement. This is similar to the use of iconic or representational gestures to represent a concrete idea. In mathematics, they are used to simulate an action or perception (Martha W. Alibali and Nathan 2012) and in computing, these gestures might be useful to facilitate
communication and simulate action. Solomon et al. found that participants who used pointing gestures rather than representational gestures for abstract concepts talked about their code at a low level, suggesting that students' gestures matched the abstract level of their understanding (A. Solomon et al. 2018). Further work could investigate how different types of gestures might help improve students' learning of abstract concepts gradually.

**RQ2.4 How are gestures used to represent abstract programming concepts (iteration)?**

RG-metaphorical gestures were used to represent abstract concepts. In Study 1, although the use of RG-metaphorical was limited to the iteration concept, we demonstrated the possibility of young children generating representational gestures spontaneously when discussing and completing a programming task. In our study, students represented iteration as a circular shape; for example, Figure 3.12, Figure 3.13, Figure 3.14, Figure 3.15 and Figure 3.18 suggest that the program will return to the point from where it started. This is a similar pattern to that found in Manches et al. (2019) when CS students were asked to represent iteration. Another metaphorical representation was the repetition of action blocks inside the repeat, such as in Figure 3.16. One student considered the repeat a container (see Figure 3.17), which is also related to the image schema found in Manches et al. (2019). Future work may be able to identify further patterns of gestures with more computing concepts and a larger representative sample. Additionally, in mathematics, RG-metaphorical gestures are used to reflect conceptual metaphors that underlie mathematical concepts (Martha W. Alibali and Nathan 2012). In computing, the use of this type of gesture may support learning of abstract concepts by externalising them and might highlight student misconceptions that might be difficult to perceive using other methods (A. Solomon et al. 2018). More research is needed to examine how RG-metaphorical gestures could reveal students' misunderstandings about an abstract concept and how teachers might be able to communicate with the students to correct their understanding.
3.6 Limitations

There are some limitations to the study described in this chapter. First, the evaluation focused on specific UK Key Stage 1 learning outcomes. Further work is needed to determine the extent to which each type of user interface is suited to learning other concepts in other countries’ curricula. Second, measurement of the children’s learning focused on the immediate effect only, not the extent to which children could apply the concepts they had learned in a corresponding situation. Therefore, more investigation is needed to determine whether further transfer is possible and to examine how children apply knowledge to solve real computer problems rather than a paper test. Third, a 40-minute learning session is a short intervention and a longitudinal study would be needed to determine whether the effects are the same across a longer intervention. Additionally, we explored how the frequency of spontaneous gestures supported learning and identified general gesture types. However, mathematics studies (Novack et al. 2014) have shown that the use of abstract gestures and concrete gestures can lead to differences in the depth and transfer of learning. Future work should investigate gesture types in these terms, to identify how they can be used most effectively in CS classrooms.

In addition, although CS uses metaphors extensively, such as programming data structures (stacks, queues, trees, pipes), our study was limited to the concept of iteration because of the young age of our target users. Other work could examine the role of metaphors in computer pedagogy more broadly. While a sample size of 42 children was appropriate for this context, it did not allow for conclusive statements about the effectiveness of the TUI-PR versus GUI-PR or spontaneous gestures, which would require a larger sample size. Finally, even though participants were enrolled in the same first grade class, children differ in their ability to effectively communicate and engage in turn-taking behaviours that advance the common goal based on their development. Thus, failure to learn and failure to use gestures may both reflect a particular child's younger development rather than supporting evidence of gestures leading to greater comprehension.
3.7 Conclusion

Study 1 investigated the effect of physicality on manipulating programming blocks by comparing primary school students’ (aged six-to-seven years) use of GUI-PR and TUI-PR. A difference between the students in the GUI-PR and TUI-PR groups was identified in learning outcomes and attitude change, but no significant difference was found between interface type and enjoyment. This may be because both the GUI-PR and TUI-PR have the same engaging output: namely, a physical robot. Therefore, Study 2, in the next chapter, investigated the role of physicality in the output and the programming environment (i.e., input). To examine this topic, isomorphic interfaces were used: first, a TUI-PR programming environment consisting of physical blocks that a user manipulates by hand to control a physical robot's behaviour; and second, a GUI-SR programming environment consisting of virtual blocks that a user can manipulate using a touchscreen to guide an on-screen robot.

In Study 1, the availability of physical elements (e.g., tangible programming blocks) did not appear to increase gesture frequency and no difference was found in the number of gestures used by students in the GUI-PR and TUI-PR groups, which may be because children in both groups had seen the physical robot and simulated its movement. The robot simulation was the main purpose for gesturing (42%). Given the apparent importance of gestures for learning, this suggests it is worth exploring other means to support children's use of gestures which Study 2 explores. Reproducing Study 1 with different outputs may deepen our understanding of how to support student gestures effectively and how this support can be implemented in an environment for learning programming. Therefore, Study 2 offers an in-depth comparison of the role of physicality in influencing learning outcomes, enjoyment, attitudes and frequency of gestures, as well as exploring the role of spontaneous gestures impacting learning outcomes, the types and purposes of gestures and the use of gesture in representing abstraction.
CHAPTER 4

STUDY 2: COMPARING TANGIBLE AND GRAPHICAL PROGRAMMING ENVIRONMENTS FOR PRIMARY SCHOOL CHILDREN

4.1 INTRODUCTION

Through a study carried out with primary school children aged six-to-seven in Saudi Arabia, Study 1 investigated the role of physicality in affecting learning outcomes, attitudes toward computing, enjoyment of computing and the number of spontaneous gestures used. The study involved comparing the children’s use of different block programming environments, a TUI and a GUI, to control a physical robot. Use of the GUI was associated with improved learning outcomes, but the TUI led to greater attitudinal improvement toward computing. No difference was found in terms of level of enjoyment, which may be because both the TUI and GUI had the same engaging physical output (i.e., controlling a robot). Also, no difference was observed in the number of gestures made by participants in the TUI and GUI groups. This could be due to the fact that both the TUI and GUI users were able to see a physical robot and thus found it natural to simulate its movement through gestures, which was the most common purpose of gesturing (42%).

For this reason, Study 2 repeated the same general method as Study 1 but changed the GUI condition to (GUI-SR) to involve an on-screen robot (SR) in comparison with the use of physical blocks to control a physical robot (PR) under the TUI condition (TUI-PR). This allowed for investigation of the role of physicality more broadly in primary programming. The study thus compared the use of a TUI-PR and a GUI-SR to teach programming skills to primary school children aged six-to-seven, using a different set of participants compared to Study 1.

In Study 1, several types of spontaneous gestures used by children were identified, including representational metaphorical gestures (RG metaphorical). In RG metaphorical, conceptual metaphors are used to represent an abstract concept (in this case, the concept
of iteration). An association was found between the frequency of gestures and learning gains. Therefore, Study 2 focused on examining the types of gestures and conceptual metaphors, as well as their association with learning, when physicality is present in only one condition (i.e., the TUI rather than the GUI). Study 2 enabled a more robust investigation of the role of physicality, which was the focus of RQ1: ‘What are the key differences in how TUI and GUI programming environments support the development of programming skills for children aged six-to-seven years old?’ particularly:

- **RQ1.1**: Are there any differences in learning outcomes, attitudes toward computing and enjoyment of computing?
- **RQ1.2**: Are there any differential gender effects in the above measures?

Moreover, Study 2 provided insights into RQ2 regarding the ways that young children, especially those aged six-to-seven, used spontaneous gestures when learning to program. In particular, Study 2 offered insights into the effect of:

- **RQ2.1**: How does interface type affect the use of children’s spontaneous gestures while completing programming tasks?
- **RQ2.2**: How are young children's learning outcomes related to the frequency of spontaneous gestures while completing programming tasks?
- **RQ2.3**: What are the types and purposes of young children's spontaneous gestures while completing programming tasks?
- **RQ2.4**: How are gestures used to represent an abstract programming concept (iteration)?

### 4.2 Methods

The study used a mixed methods and between-groups research design. For the first research question, the independent variables were (i) interface type (TUI-PR and GUI-SR) and, (ii) gender. The independent variable (IV) for RQ2.1 was (i) interface type and for RQ2.2 the IVs were frequency of gesture (high or low). The participants completed a pre-test and pre-attitudinal survey; a learning session, which included programming a
robot; a post-test; and attitudinal and enjoyment surveys. The learning session was recorded, and each participant’s spontaneous gestures was identified and coded. The participants were not encouraged to gesture. When children used their hands or body to debug their code and/or tried to comprehend programming tasks or explain the task, their gestures were recorded and coded. Given that previous research had suggested that teachers’ use of gestures can benefit student learning (Church, Ayman-Nolley, and Mahootian 2004; Valenzeno, Alibali, and Klatzky 2003), researcher gestures – if any – were recorded and coded to check for any difference in gesture frequency between the groups. The dependent variables for RQ1, RQ2 were the following: 1) normalised learning gain (calculated by the difference between pre- and post-test scores); 2) attitudinal change (calculated by the difference in pre- and post-activity attitudinal survey scores); 3) enjoyment (calculated by scores on enjoyment survey) and 4) number of gestures performed by participants; 5) gesture types; 6) number of gestures; 7) session time (from the video); 8) interaction time; and 9) number of attempts (from application log data).

4.2.1 Participants and Setting

The study was conducted at an International primary school in Saudi Arabia, but with a different set of participants compared to Study 1, described in the previous chapter. A pre-test and pre-attitudinal survey were completed in the participants’ classrooms. A quiet room was used for the learning activities, post-tests, post-surveys and enjoyment surveys. Five classes, all single-gender, were grouped into single-gender pairs, apart from one mixed pair, because there was an odd number in both of the single-gender groups. A total of 70 children (25 females and 45 males) aged between six-to-seven years were invited to participated in the study. Of these children, only 50 children’ parents gave their informed consent for their child to participate in the study. The 50 participants were paired with same-gender children based on them having similar teacher-assessed levels in mathematics, based on the relation between mathematics abilities and computing skills (Wing 2010) see Table 4.1. Pairs of participants were randomly assigned to conditions. Initially, the learning session for 4 pairs of participants took place in the school’s science lab. However, the room setup prevented movement and affected the children’s
spontaneous gestures. The high stools used as seating the lab appeared to be inhibiting children’s spontaneous gestures, perhaps because they were using their hands to ensure they stayed balanced. Therefore, these data were deleted from the study. To allow participants to move more freely, we relocated the study to an empty classroom, see Table 4.1 in which the remaining 21 pairs of participants completed their sessions. The distribution of participants across the conditions is GUI (N=20, pairs= 10) and TUI (N=22, pairs=11), which N is the number of participants.

Table 4.1 Group characteristics

<table>
<thead>
<tr>
<th></th>
<th>GUI-F</th>
<th>GUI-M</th>
<th>GUI-FM</th>
<th>TUI-F</th>
<th>TUI-M</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>6</td>
<td>12</td>
<td>2</td>
<td>10</td>
<td>12</td>
<td>42</td>
</tr>
<tr>
<td>Pairs</td>
<td>3</td>
<td>6</td>
<td>1</td>
<td>5</td>
<td>6</td>
<td>21</td>
</tr>
<tr>
<td>M&amp;S M</td>
<td></td>
<td></td>
<td>93.3%</td>
<td>85.8%</td>
<td>85%</td>
<td>84%</td>
</tr>
<tr>
<td>M&amp;S SD</td>
<td></td>
<td></td>
<td>8.1</td>
<td>5.1</td>
<td>7.0</td>
<td>5.1</td>
</tr>
</tbody>
</table>

FM is for a group that has female and male participants because the same sex groups were odd numbers

Figure 4.1 Study location

4.2.2 Materials

The same pre- and post-attainment tests, attitudinal surveys, post-activity enjoyment survey, GoPro video camera for data collection, activity sheet that was used in the learning
activity and TUI-PR were used in Study 1. The new update was in the GUI interfaces as it consisted of an android application with a graphical programming block and on-screen robot.

**SYSTEM DESCRIPTION**

The same system that was used in study 1 was used including the TUI-PR see Figure 4.2 but the GUI was updated to fit this study objective see Figure 4.3.

![Figure 4.2 TUI setting (left) and mBot robot and robot mat (right)](image1)

![Figure 4.3 GUI programming blocks (left) and on-screen robot (right)](image2)
In particular, the GUI blocks on the tablet’s touchscreen were used to program on screen robot instead of the physical robot. Similar to the physical robot, the on-screen robot moved on a screen grid (5×5 grid), which had four types of tiles to configure the tasks, see Figure 4.4.

**THE TEST AND SURVEYS**

The same attainment test, attitudinal and enjoyment surveys were used as for Study1 (see 3.2.2.4-3.2.2.6) and Appendix A and B for full details)

**LEARNING ACTIVITIES**

In Study 1, children completed six activities with different difficulty levels. However, the total session length, which was 45 minutes, was considered too long for children of this age, as was reported in Chapter 3. Therefore, In Study 2, the number of activities was reduced to four and only simple tasks were selected to reduce the time to 35 minutes, thus excluding Activities 3 and 6, see Figure 4.5.
<table>
<thead>
<tr>
<th>Activity 4</th>
<th>Activity 5</th>
<th>Activity 6 (Moderate Complexity /Removed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 Forward</td>
<td>Repeat {Left, Forward}</td>
<td>Repeat {Forward, Right}, Forward</td>
</tr>
<tr>
<td>Right, Forward, Noise, Left, Forward</td>
<td></td>
<td>Repeat {Forward, Noise, Forward}</td>
</tr>
</tbody>
</table>

**Figure 4.5 Learning activities**

**4.2.3 Procedure**

Two sessions were held with the participants between November and December of 2019. The procedure consisted of first session and second session similar to study 1. In the first session the researcher introduced herself and, distributed the pre-survey and the pre-test, the session last approximately 20 minutes. The second session took place in a empty classroom in the school and lasted approximately 35 minutes. Pairs of participants finished the 4 learning activities together, followed by an individual post-test, post-attitudinal test and an enjoyment survey. The participants were asked to solve Activities 1, 2 and 4. The repeat block functionality was then explained, and the participants were asked to solve Activity 4 using a different method and then Activity 5. If they solved the activity without using a repeat, the researcher would ask them to think about another solution. For more details about the procedure see Section 3.2.4.
4.3 Analysis

4.3.1 Test Schemas

To analyse participants’ answers to the open-ended questions (Q3–Q6) and participants’ errors in an attempt, the coding scheme from Study 1 was used (see Section 3.3.1). The author coded the test data and then a random sample of 20% of the answers to Questions 4 and 5 was generated for second coding which was conducted by a Masters-level student. Cohen’s kappa was used as an inter-rater reliability test with a kappa value of 0.946.

4.3.2 Overview of Gesture Coding Rubrics

21 sessions were conducted but 3 video recording were missing due to a technical error with the camera (2 TUI and 1 GUI). 18 videos of learning activities were coded with a mean duration of 21 minutes, see Table 4.2 for the total distribution of participants across conditions.

<table>
<thead>
<tr>
<th></th>
<th>GUI-F</th>
<th>GUI-M</th>
<th>GUI-FM</th>
<th>TUI-F</th>
<th>TUI-M</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>N</strong></td>
<td>6</td>
<td>10</td>
<td>2</td>
<td>8</td>
<td>10</td>
<td>36</td>
</tr>
<tr>
<td><strong>Pairs</strong></td>
<td>3</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>18</td>
</tr>
<tr>
<td><em>FM is for a group that has female and male participants</em></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The coding rubric for type, location and purpose from Study 1 was used (see Section 3.3.2). After the first author had coded all the video data, a random sample of 20% was generated for second coding by PhD student. Inter-rater reliability was determined using Cohen’s kappa scores, with strength agreement based on Fleiss (Fleiss, Levin, and Paik 2004) (the reliabilities were: for Type = 0.714, moderate; for Location = 0.827, strong; and for Purpose = 0.646, moderate).

4.4 Results

This section presents the results that contribute to RQ1: ‘When comparing TUIs and GUIs for programming activities with children aged six-to-seven, how does interface type influence children’s learning outcomes, attitudes towards computing and enjoyment of programming activities? What are the differential gender effects in the above measures?’ (Section 4.4.1 and Section 4.4.2) and RQ2: ‘How do young children use spontaneous
gestures while learning programming, and what is the relationship between gesturing and learning outcomes?’ (Section 4.4.3).

4.4.1 Effect of Output Physicality and Gender on Learning Outcomes, Attitudes and Enjoyment

This section presents the results of the attainment tests and surveys to answer the RQ.1: ‘What are the key differences in how TUI and GUI programming environments support the development of programming skills for students aged 6-7 years old?’ RQ1.1 ‘How does interface type TUI-PR and GUI-SR influence children’s learning outcomes, attitudes towards computing and enjoyment of programming activities?’ ‘RQ1.2 what are the differential gender effects in the above measures?’

Interface Type and Learning Outcomes

Normalised learning gains were used to measure learning outcomes. These were calculated by considering the maximum possible gain or loss given the pre-test score (Marx and Cummings 2007), using the formula: $100 \times \frac{\text{post}-\text{pre}}{100-\text{pre}}$. An independent samples t-test was performed to determine whether interface type impacted children’s normalised learning gains after the test’s assumptions had been satisfied. TUI users showed slightly greater normalised learning gains on average ($M=10.7$, $SD=5.1$) compared to the GUI users ($M=9.1$, $SD=5.2$), but the difference was not statistically significant (95% CI, -1.60 to 4.8), $t(40) = 1.022$, $p = .313$).

Gender and Learning Outcomes

An independent samples t-test was performed to determine whether gender impacted children’s normalised learning gains after the test’s assumptions had been satisfied. Male users showed greater normalised learning gains on average ($N=25$, $M=10.1$, $SD=5.7$) compared to the female users ($N=17$, $M=9.6$, $SD=4.3$), but the difference was not statistically significant (95% CI, -3.81 to 2.86), $t(40) = -288$, $p = .775$).

Interface Type and Gender and Learning Outcomes

The average normalised learning gains for the groups by gender for (M, TUI) is ($M=10.71$, $SD = 6.95$), (F, TUI) ($M=9.67$, $SD=3.7$), (M, GUI) ($M=8.58$, $SD=5.03$) and (F, GUI)
(M=8.63, SD=5.34). A two-way ANOVA was run to examine the effects of interface and gender. The interaction effect between interface and gender on normalised learning gains was not statistically significant F (1, 38) = .110, p = .742, partial η² = 2.93.

**INTERFACE TYPE AND ATTITUDINAL CHANGE**

Analysis of the before and after attitudinal survey data was conducted on its five questions each with five categories, so it could be used as a continuous data (see Appendix B). The data were aggregated into a single attitudinal measure on a 25-point maximum and 5-point minimum scale. An independent samples t-test was performed to determine whether interface type impacted children’s attitude toward computing. The GUI group showed a greater increase in scores (M=2.10, SD=1.94) compared to the TUI group (M=0.27, SD=2.21), with a statistically significant difference in mean attitudinal change scores between the two interfaces (95% CI, [0.313, -0.52], t (40) = 2.834, p = 0.007).

**GENDER AND ATTITUDINAL CHANGE**

An independent samples t-test was performed to determine whether gender impacted children’s attitudinal change after the test’s assumptions had been satisfied. Male users showed greater normalised learning gains on average (N=25, M=10.1 SD=5.7) compared to the female users (N=17, M=1.48, SD=2.14), but the difference was not statistically significant (95% CI, -2.61 to .595), t (40) = -1.17, p = .257).

**INTERFACES GENDER ON ATTITUDINAL CHANGE**

The average attitudinal change for the groups by gender was (M, TUI) (M= .92, SD = 2.35), (F, TUI) (M= .50, SD=1.84), (M, GUI) (M=2.0, SD=1.87) and (F, GUI) (M= 2.29, SD=2.21). A two-way ANOVA was conducted to examine the effects of interface and gender. The interaction effect between interface and gender on attitudinal change scores was not statistically significant F (1, 38) = 1.68, p = .203, partial η² = 7.19.

**INTERFACE TYPE AND ENJOYMENT**

Since the enjoyment survey consists of the ordinal data (see Appendix B) and data were aggregated into single enjoyment measure on a 11-point maximum and 3-point minimum scale. The Mann-Whitney U test was performed to determine whether interface type
impacted children’s enjoyment. The TUI group showed a slightly greater increase in scores (N=22, Mean Rank= 22.27) compared to the GUI group (N=20, Mean Rank= 20.65), but no statistically significant difference was observed in enjoyment scores between the two interfaces (U=237, z=.495, P=0.621).

**GENDER ON ENJOYMENT**

Additionally, A Mann-Whitney U test was run to determine if there were differences in enjoyment score between gender. Enjoyment scores for females (N=17, mean rank = 19.71) and males (N=25, mean rank = 22.72) were not statistically significantly different, U = 243, z = .904, p = .366.

Summary diagrams with the tests and surveys by the condition can be seen in Figure 4.6.

![Figure 4.6 Summary of results by condition (a) Learning gain; (b) Attitudinal change; Enjoyment score.](image)

**4.4.2 Overview of Learning Session Data**

This subsection presents an overview of the quantitative data collected for 18 groups from the video recording (session time) and the application log data (interaction time and number of attempts) see Table 4.3 and Table 4.4. The session time is the video recording duration, which serves as a close proxy for overall session time (as the recording was started just before the researcher gave an overview of the learning session and ended directly after the fourth learning activity finished).
Whenever a participant executed their program with the play button, the number of
attempts was defined. The interaction time was calculated as the total time across all
attempts that the pairs spent interacting with the programming blocks (from the start of
their attempt until they clicked the play button to test their code).

Table 4.3 Overall descriptive session data for the two condition groups (N is the number of pairs)

<table>
<thead>
<tr>
<th></th>
<th>TUI</th>
<th>GUI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>N  M  SD</td>
<td>N  M  SD</td>
</tr>
<tr>
<td>Session time</td>
<td>9 28 mins 4.5</td>
<td>9 22.4 mins 5</td>
</tr>
<tr>
<td>Interaction time</td>
<td>9 7 mins 2.2</td>
<td>9 11 mins 4.3</td>
</tr>
<tr>
<td>Number of Attempts</td>
<td>9 12 2.9</td>
<td>9 15 3.4</td>
</tr>
</tbody>
</table>

Table 4.4 Overall descriptive session data for the two condition groups divided by gender (N is the number of pairs)

<table>
<thead>
<tr>
<th></th>
<th>TUI</th>
<th></th>
<th></th>
<th>GUI</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>F</td>
<td>M  SD</td>
<td>M  SD</td>
<td>M</td>
<td>M  SD</td>
<td>M  SD</td>
</tr>
<tr>
<td>Session time</td>
<td>4</td>
<td>29.7 mins 6.2</td>
<td>5 26.7 mins 2.6</td>
<td>3</td>
<td>22.6 mins 8.2</td>
<td>5 22.6 mins 4.0</td>
</tr>
<tr>
<td>Interaction time</td>
<td>4</td>
<td>8.7 mins 2.5</td>
<td>5 6.5 mins 1.6</td>
<td>3</td>
<td>13.3 mins 8.3</td>
<td>5 10.9 mins 1.7</td>
</tr>
<tr>
<td>Number of Attempts</td>
<td>4</td>
<td>13 1.8 mins 5</td>
<td>11.8 3.9</td>
<td>3</td>
<td>13 1.8 mins 5</td>
<td>14.6 3.6</td>
</tr>
</tbody>
</table>

**EFFECT OF INTERFACE ON SESSION TIME**

An independent samples t-test was performed to determine differences in session time
between interface types after the test’s assumption had been satisfied. The TUI group
spent more time (N=9, M=28.05, SD=4.5) than the GUI users (N=9, M=22.43, SD=5) and
the difference was statistically significant (95% CI, .82 to 10.4), t (16) =2.48, p =.024).

**EFFECT OF GENDER ON SESSION TIME**

The average session time for the groups by gender for male was (N=10, M=24.6, SD =
3.8), female (N=7, M=26.8, SD=7.51 An independent samples t-test was performed to
determine differences in session time between gender after the test’s assumption had been
satisfied. The female group spent more time than the male users, but the difference was
not statistically significant (95% CI, -3.8 to 7.95), t (15) =740, p =.471).
EFFECT OF GENDER AND INTERFACES ON SESSION TIME

The average session time for the groups by gender for (M, TUI) was (M=26.7, SD = 2.6), (F, TUI) (M=29.22, SD=6.22), (M, GUI) (M=8.58, SD=5.03) and (F, GUI) (M=22.6, SD=4.04). A two-way ANOVA was run to examine the effects of interface and gender. The interaction effect between interface and gender on session time was not statistically significant F (1,13) = .006, p = .937, partial $\eta^2$ = .096.

EFFECT OF INTERFACE ON INTERACTION TIME WITH PROGRAMMING ENVIRONMENT

The TUI group had a significantly shorter interaction time (N= 9, M=7, SD=2.2) than the GUI group (N=9, M=11, SD=4.3). An independent samples t-test was performed to check for any difference between interaction time and interface type after the independent t-test’s assumptions had been satisfied. A statistically significant difference was identified (95% CI, -7.64 to -.309), t (16) = -2.99, p =.023).

EFFECT OF GENDER ON INTERACTION TIME

The average interaction time for the groups by gender for male was (N=10, M=8.7, SD = 2.8), female (N=7, M=10, SD=5.7) An independent samples t-test was performed to determine differences in interaction time between gender after the test’s assumption had been satisfied. The female group spent more time than the male users, but the difference was not statistically significant (95% CI, -2.7 to 2.1), t (15) =.830, p =.420).

EFFECT OF GENDER AND INTERFACES ON INTERACTION TIME

The average session time for the groups by gender for (M, TUI) is (M=6.7, SD = 2.6), (F, TUI) (M=8.7, SD=2.5), (M, GUI) (M=10.99, SD=1.74) and (F, GUI) (M=13.1, SD=8.3). A two-way ANOVA was run to examine the effects of interface and gender. The interaction effect between interface and gender on session time was not statistically significant F (1,15) = .000, p = .994, partial $\eta^2$ = .001.

EFFECT OF INTERFACE ON NUMBER OF ATTEMPTS

An independent samples t-test was performed to check for any difference between interface type and the number of attempts. The test’s assumption was satisfied. There was
a trend towards the GUI group needing more attempts to complete the tasks successfully (N=9, M=14, SD=2.9) compared to the TUI group (N=9, M=12, SD=2.8). However, no statistically significant difference was observed (95% CI, -5.58 to .916), t(16) = -1.52, p = .160).

**EFFECT OF GENDER ON NUMBER OF ATTEMPTS**

The average on number of attempts for the groups by gender for male was (N=10, M=13.6, SD =), female (N=7, M=13.4, SD=5.7) An independent samples t-test was performed to determine differences on number of attempts between gender after the test’s assumption had been satisfied. The male group spent more time than the female users, but the difference was not statistically significant (95% CI, 1.71 to 3.20), t (15) = -.100, p =.922).

**EFFECT OF GENDER AND INTERFACES ON NUMBER OF ATTEMPTS**

The average session time for the groups by gender was for (M, TUI) (M=11.8, SD = 3.9), (F, TUI) (M=13.00, SD=1.8), (M, GUI) (M=14.6, SD=3.6) and (F, GUI) (M=13.33, SD=3.7). A two-way ANOVA was run to examine the effects of interface and gender. The interaction effect between interface and gender on session time was not statistically significant F (1,15) = .626, p = .441, partial η2 = .441.

**4.4.3 Use of Spontaneous Gestures While Solving Programming Tasks**

This section presents the result of the gestural analysis for the 18 videos session in order to answer the RQ2: ‘How do young children use spontaneous gestures while learning programming, and What is the relationship between gesturing and learning outcomes?’

**RQ2.1 HOW DOES INTERFACE TYPE AFFECT CHILDREN’S USE OF SPONTANEOUS GESTURES WHILE COMPLETING PROGRAMMING TASKS?**

An independent samples t-test was performed to determine whether interface type affected the frequency of children’s spontaneous gestures. The mean number of gestures produced by the GUI group (M=11.1, SD=1.22) was lower than that produced by the TUI group (M=12.7, SD=1.7), but the difference was not statistically significant. An independent samples t-test was performed to ensure that the researcher’s gestures had not influenced the participant gestures and also to ensure that the researcher did not gesture more with
one group compared to another. No difference was observed in the frequency of researcher gestures across the interface types. The researcher’s mean number of gestures was slightly higher in the TUI group (M=33.9, SD=3.04) compared to the GUI group (M=31.44, SD=9.94). The data had an acceptable range of normality, with z values in the range of +1.96, which for a small sample size (N < 50) is considered normally distributed (Kim 2013; Pett 2016). Researcher gestures were considered because teacher gestures might benefit learners (Church, Ayman-Nolley, and Mahootian 2004; Valenzeno, Alibali, and Klatzky 2003).

**RQ2.2 HOW ARE YOUNG CHILDREN’S LEARNING OUTCOMES RELATED TO THE FREQUENCY OF SPONTANEOUS GESTURES WHILE COMPLETING PROGRAMMING TASKS?**

In Study 1 an association between spontaneous gestures and learning was found see Section 3.4.3.2 in Chapter 3. We continue with the question of whether there were participants who could be characterised as either ‘high gesturers’ or ‘low gesturers, as well as checking for the existence of a relationship between gesture frequency and learning gain as it was done in Study 1. Therefore, the dataset was divided by quartile according to the frequency of gesture, characterizing participants in the upper quartile as ‘high frequency gesturers’ and those in the lower quartile as ‘low frequency gesturers’. An independent samples t-test was performed to determine if there was a significant difference between the attainment pre-test scores. The scores for the attainment pre-test for the ‘high gesturers’ group was (M=5.32, SD=3.30) and for the ‘low gesturers’ group was (M=5.07, SD=3.56) indicated that the difference was not statistically significant.

The normalised learning gains score for the ‘high frequency gesturers’ was (M= 12.32, SD=1.4) and for ‘low g frequency gesturers’ was (M=10.16, SD=1.8), but not statistically significant. Using an independent samples t-test, there was no statistically significant difference in learning gains between the high-frequency gesturing and low-frequency gesturing groups (t (16) =921, p=.371).

The results might have been affected by the number of researcher gestures or the complexity of the activity. First, this research was exploratory and the researcher’s gestures were not controlled, to mirror CS instructors in real classrooms (A. Solomon et
al. 2020), therefore seeing the researcher gesturing might prompt students to gesture. The researcher gestures were quantified the mean number of gestures for the participant (P) and researcher (R) in ‘high frequency gesturers’ was (P= 20.56, SD=.988, R= 24.56, SD=4.9), while for ‘low frequency gesturers’ it was (P=4.00, SD=0.408, R=36.56, SD= 13.17). An independent samples t-test was undertaken to determine whether there was a difference between the researcher’s gestures in each frequency group. A statistically significant difference was identified (t (13) =-2.55, p=0.028)). The results indicates that there was difference in the number of researcher gestures between groups, which may have influenced students’ gestures. This point was discussed in section 4.5.2.

An additional analysis of the number of gestures for each activity in both Study 1 and 2 is described in the next subsection that checked the effect of the activity complexity on learning outcomes.

**RELATIONSHIP BETWEEN GESTURE AND PROBLEM COMPLEXITY**

In Study 1, we found a difference between the high and low frequency gesturer groups and learning gains, but this was not the case in Study 2. This disparity prompted some additional analysis to support investigation of possible differences in the use of gesturing across the studies. One difference in design between the two studies was the outputs of the interfaces, but there was no significant difference in gesture frequency between the TUI and GUI in either study. The second difference in design was the number and complexity of the activities. Therefore, this section presents an analysis of the number of gestures for each activity in both Study 1 and Study 2, as activity complexity could be the factor that prevented Study 2 from having the same result as Study 1 in this area of investigation.

The activity complexity was estimated based on the intrinsic cognitive demands of the activity, which was high for Activities 3 and 6, see Section 3.2.3 for more detail about the activity complexity.

In Study 1, we included four simple activities (Activities 1, 2, 4 and 5) and two complex activities (Activities 3 and 6) for participants to complete and in Study 2 the complex...
activities (3 & 6) were removed to decrease the overall time of the session (see Figure 4.5 Learning activities).

The tables show breakdown of number of gestures, session time and gesture per minute for each activity see Table 4.5 for study 1 and Table 4.6 for study 2.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Activity</th>
<th>Activity</th>
<th>Activity</th>
<th>Activity</th>
<th>Activity</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple</td>
<td>Simple</td>
<td>High</td>
<td>Simple</td>
<td>Simple</td>
<td>Moderate</td>
<td></td>
</tr>
<tr>
<td>Number of gestures</td>
<td>54</td>
<td>168</td>
<td>331</td>
<td>104</td>
<td>199</td>
<td>163</td>
</tr>
<tr>
<td>Mean session time</td>
<td>1.7 mins</td>
<td>5.6 mins</td>
<td>8.1 mins</td>
<td>6.4 mins</td>
<td>4.2 mins</td>
<td>5.1 mins</td>
</tr>
<tr>
<td>Gestures per minute</td>
<td>1.7</td>
<td>1.6</td>
<td>2.3</td>
<td>0.9</td>
<td>2.5</td>
<td>1.7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Activity</th>
<th>Activity</th>
<th>Activity</th>
<th>Activity</th>
<th>Activity</th>
<th>Activity</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple</td>
<td>Simple</td>
<td>High</td>
<td>Simple</td>
<td>Simple</td>
<td>Moderate</td>
<td></td>
</tr>
<tr>
<td>Number of gestures</td>
<td>56</td>
<td>86</td>
<td>-</td>
<td>162</td>
<td>124</td>
<td>-</td>
</tr>
<tr>
<td>Mean session time</td>
<td>1.8 mins</td>
<td>4.5 mins</td>
<td>-</td>
<td>8.0 mins</td>
<td>6.2 mins</td>
<td>-</td>
</tr>
<tr>
<td>Gestures per minute</td>
<td>1.6</td>
<td>1.0</td>
<td>-</td>
<td>1.1</td>
<td>1.1</td>
<td>-</td>
</tr>
</tbody>
</table>

In Study 1, participants generated the highest number of gestures in Activity 3, which was a complex task and required a high cognitive level of thinking see Table 4.6. Given the previous assumption about activity complexity, we re-examined the data from Study 1,
excluding the complex activities (Activity 3 and Activity 6 gestures) that used the same activities used in Study 2. The mean number of gestures for ‘high-frequency gesturers’ was 9.15 (SD=1.6), while for ‘low-frequency gesturers,’ it was 5.5 (SD=.95). An independent sample t-test showed no statistically significant difference in learning gains between the high-frequency and low-frequency gesturing quartiles (t (12) =1.88, p=0.84). The results were similar to study 2.

However, the gesture per minutes row in Table 4.5 showed that Activity 5 had the greatest gesture per minutes rate, even though the researcher classified Activity 3 as a highly complex activity. Additionally, in Study 2, Activity 1 has the greatest gesture rate per minute even though it was classified as simple. For further interpretations of this result (see the discussion in Section 4.7.2.2).

**RQ2.3 What are the types and purposes of young children's spontaneous gestures while completing programming tasks?**

After coding the 428 gestures in Study 2, we found that the gesture type used most frequently was pointing (343), followed by representational gestures (85). In terms of location, gestures were performed most frequently on/near the activity sheet (253), with those near/above the robot mat (59), near the code (99) and in the air (17). The purposes of the gestures were simulation (37.4%), direction (17.5%), indicating objects (27.8%), counting (12.4%) and iteration (4.9%).

**Relationship between gesture type and gesture purpose**

To identify the patterns that emerged from the participants’ gesture types and purpose, we used a cross-tabulation showing the data for a given gesture’s purpose and the most frequent type of gesture used. Table 4.7 below shows the balance of pointing and representational gesture types across the five purposes identified.

**Table 4.7 Cross-tabulation for gesture type and purpose**

<table>
<thead>
<tr>
<th>Pointing (Deictic Proportion)</th>
<th>Representational Proportion</th>
<th>Total Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation</td>
<td>88%</td>
<td>11. %</td>
</tr>
<tr>
<td>Direction</td>
<td>26%</td>
<td>73%</td>
</tr>
</tbody>
</table>
Similar to Study 1, most of the purposes analysed demonstrated a simple concrete concept such as simulation of the robot movement, direction (e.g., right turn), objects, or counting. noticing young children representing abstraction spontaneously using their gestures is worthwhile. For this reason, the next subsection discusses the gestures used by the participants to represent the concept of iteration.

RQ2.4 How are gestures used to represent an abstract programming concept (iteration)?

Across the 36 participants for whom we had video data, only 8 performed gestures that represented the abstract concept of iteration. Pointing to the repeat block was not included in this analysis. Instead, we were interested in investigating the use of representational gestures for abstract concepts which produced 10 gestures. Table 4.8 shows the categories of the representational gestures of the concept ‘repeat’ as, 7 drawing a circle in the tablet and 1 performing the action inside the repeat twice. The representational gestures were exhibited by each participant while solving the programming task. No significant differences in learning gains were found between the participants who used gestures representing an abstract concept and their counterparts.

<table>
<thead>
<tr>
<th>Description of gesture</th>
<th>Image example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drawing a circle on the tablet with the pointing finger</td>
<td>Figure 4.7 P01</td>
</tr>
</tbody>
</table>

Table 4.8 Iteration representational gesture description
4.5 DISCUSSION

This study presented an in-depth comparison of the use of TUI/physical robot and GUI/on screen robot to teach programming skills to primary school children aged six-to-seven years. Using a mixed methods research design, we conducted a comparative study of two similar interfaces, the only difference being the physicality element. Study 2 contribute to answer RQ1 and RQ2 in the following sections.

4.5.1 RQ1 What are the Key Differences in How TUI And GUI Programming Environments Support the Development of Programming Skills for Students Aged 6-7 Years Old?

To explore the key differences in how TUI and GUI support the development of programming skills for young children (RQ1), the question is broken into four sub questions (RQ1.1 and RQ1.2).

RQ1.1 Effect of output physicality on learning outcomes, attitudes and enjoyment of computing

There is promising research on the relevance of embodied cognition in non-CS areas indicating that physical manipulation can benefit learning in reading comprehension (Glenberg, Goldberg, and Zhu 2010) and science (Kontra et al. 2015). However, there is still no evidence of this in the field of CS. Our finding is in line with a previous study on the effects of interface type on learning outcomes (Strawhacker and Bers 2015). The study reported no significant difference in learning gains between groups, despite their slight differences in the role of physicality. However, this contrasted with the findings in Study 1, which found that GUI users experienced significantly greater learning gains than TUI
users did when both interfaces program a physical robot. In terms of attitude toward computing measures, the GUI group experienced a significantly greater improvement in attitude. This result in Study 2 might be explained by the fact that the children spent more time interacting with the interface in the GUI group and less session time compared to TUI. The session time, which included the time taken to understand the task, plan the solution, debug and execute the code, was significantly higher in the TUI group compared to the GUI. The TUI users needed more session time due to the additional time needed to set up the robot mat and for the physical robot to move. In terms of the number of attempts participants needed to accomplish the tasks, the GUI group took more attempts than the TUI group. Although the difference, in this case, was not statistically significant and a similar result was found in (Sapounidis and Demetriadis 2013; Xie, Antle, and Motamedi 2008).

Despite the difference in attitude change and the use of a graphical robot and tangible robot as different engaging outputs, no significant difference was in enjoyment. Both groups enjoyed the interfaces at similar levels to the report in Study 1 and the children enjoyed the short activities with the robot

**RQ1.2 Are there any differential gender effects in the measures?**

There were no statistically significant differences between genders and interfaces for all three dependent variables, Additionally, no statistically significant differences between genders and three dependent variables, in line with the first study.

**4.5.2 RQ2 How Do Young Children Use Spontaneous Gestures While Learning Programming and What is the Relationship Between Gesturing and Learning Outcomes?**

To explore how children used their spontaneous gestures (RQ2), the question is broken into four sub-questions (RQ2.1- RQ2.4).
RQ2.1 How are young children’s learning outcomes related to the frequency of spontaneous gestures while completing programming tasks?

No difference was observed in the number of gestures between interface types, but the TUI group generated more gestures than the GUI group. A statistically significant difference was not found between the high and low gestures groups in learning outcomes, in contrast to Study 1.

RQ2.2 How are young children’s learning outcomes related to the frequency of spontaneous gestures while completing programming tasks?

In Study 1, the learning activity consisted of four simple and two complex activities for participants to complete. They produced 1,019 gestures in total while, in Study 2’s learning activities, participants generated 428 gestures across the four simple activities. In Study 1, around one-half of the gestures were generated when participants performed the two complex activities removed in Study 2 to reduce the overall session time from 45 minutes to 35 minutes based on school requests. Therefore, it might be the case that the activity’s complexity is the factor that drives learners to use gestures that influence their learning. Gestures may be used as a tool to offload cognitive load (M. Wilson 2002) and to assist learners in getting high learning gains. Therefore, the participants did not need to use such a tool because the activities were simple and did not require complex cognitive work.

However, the number of gestures generated per minute did not accord with the estimated complexity of the activities, where activity complexity was defined on the basis of the inherent cognitive demand of the activity (i.e. problem difficulty).

Rather, the results indicate two possible explanations. The first possibility is that gesture frequency is not related to activity complexity, irrespective of how it is measured (i.e., gesturing is not used as an offloading tool). Further research may extend this investigation and explore the role of gesture frequency and its relationship to activity complexity.

The second explanation is that gesture frequency is related to activity complexity but the measurement of activity complexity must be adjusted. Activity complexity can be
redefined as a function of multiple factors, including problem difficulty, the individual’s level of familiarity with the utilised concepts and the time taken to complete the activity.

In Activity 1, the participants were asked to move the robot four steps forward. In Study 1, the number of gestures per minute (1.7) was equal to that for Activity 6, which was regarded as a more complex activity. In Study 2, Activity 1 was associated with the highest number of gestures per minute (1.6). Although this activity was the simplest from the standpoint of problem difficulty, the participants at this point were not familiar with the system and commands. In further research, a fruitful area of investigation is the role of gestures as a tool to offload cognitive load. Researchers could examine this topic by analysing learner gestures when solving programming activities of different complexity levels and the analysis could span several sessions rather than a single intervention. This would help to distribute the learning times and the ability of the participants to understand the computing concepts. It may also show how gestures help learners and how gestures are influenced by activity complexity.

Also, a statistically significant difference ($t (13) =-2.55, p = 0.028$) was observed in the number of gestures produced by the researcher between the conditions (high and low gestures groups). This may have influenced the results because it is possible that the researcher’s gestures encouraged participants to gesture. That is to say, the gestures used by the participants may not have been the product of their aim of offloading the cognitive load of the programming task; instead, the participants might have been imitating the researcher. Further work might investigate the impact of teachers gestures on children’s gestures when they are solving programming tasks.

RQ2.3 What are the types and purposes of young children’s spontaneous gestures while completing programming tasks?

We examined the different types of gestures that children produce when solving programming tasks. Pointing, which was used to reflect the grounding of cognition in the environment, was the most used gesture by the participants to indicate an object. McNeill indicated that pointing gestures physically link speech and associated mental processing gestures to the physical environment (McNeill 1994). This was found in our study when
the participants used it to simulate the code and was also evidence in Solomon et al.’s report of deictic gestures assisting participants in tracing the code (A. Solomon et al. 2018). Finally, pointing represents a simple illustration of grounded cognition theory, namely that tools from the environment were used to offload cognitive work (M. Wilson 2002). For example, when the research participants counted the number of squares, they wanted the robot to walk in the correct direction or help participants understand its direction. The iconic gesture is a representational gesture that represents some concrete (McNeill 1994). It was the second most frequent type of gesture used in this study’s sample of participants. In mathematics, such gestures were used to simulate an action or perception (Martha W. Alibali and Nathan 2012). In this research context, iconic gestures were used mostly to indicate direction (e.g., turning or moving forward). Iconic gestures can be helpful to facilitate communication as well as simulate action (A. Solomon et al. 2018).

**RQ2.4 HOW ARE GESTURES USED TO REPRESENT AN ABSTRACT PROGRAMMING CONCEPT (ITERATION)?**

Finally, representational gestures were used to represent iteration, which is an abstract concept. We introduced one abstract concept to the participants because it is suitable for participants’ young age, as the literature suggested (Barefoot 2018). We demonstrated the possibility that young children could produce representational gestures spontaneously when they were solving programming problems. We noticed that children mostly produced these gestures when they were explaining the repeat concept to the researcher before they redid Activity 4. This is interesting as it is an indication that computing concepts were embodied and children used a conceptual metaphor when they were explaining ‘iteration’, similar to the adults in another piece of research (Manches et al. 2020). Metaphors were used to provide ways (e.g., visual imagery) to help pupils to access abstract notions (Manches et al. 2020). For example, many programming environments such as scratch have adopted block building as a visual metaphor to help children understand the syntactic nature of code and physical activities have been aimed to engage children with body-based analogies of computing concepts, such Unplugged (see Chapter 2). According to Cortina in 2015 such physical activities were effective because ‘by being
physically part of the solution to a problem as it is being solved, kids learn from observations and experiences (Cortina 2015). Additionally, representational gestures could reveal children’s understanding about an abstract concept and how teachers might be able to communicate with the children to correct their understanding (McNeill 1994). However, no clear theoretical framework is provided for why certain body-based experiences may develop thinking, beyond motivation and engagement (Manches et al. 2020).

Understanding the use of representational gestures in computing education can improve children’s ability to understand computer concepts at a high level. Solomon et al. in 2018 found that participants who used representational gestures rather than pointing gestures talked about their program at a higher-level, indicating that gestures may correspond to abstract understanding (A. Solomon et al. 2018). In our study, participants represented iteration as a circular shape. For example, Figure 4.7 P01 and Figure 4.8 P05 action inside the repeat (right) suggest that the program will return to its starting point. This is a similar pattern to that found in adult students studying CS engagement (Manches et al. 2020) when they were asked to explain programming concepts and with children (Almjally, Howland, and Good 2020b) when they were solving programming tasks. Given these results, Study 3 focused on exploring the role of conceptual metaphors of basic programming concepts such as ‘iteration’ by children, six-to-seven years old. The study explored the role of conceptual metaphors via an analysis of children’s use of spontaneous gestures when explaining programming concepts. It is contribution to examine representational-metaphorical gestures because they provide some evidence that thinking about computing is embodied (Manches et al. 2020), and may reflect understanding and map meanings that are not expressed verbally (Novack, Goldin-Meadow, and Woodward 2015). This is the first step of understanding children use of conceptual metaphor it may help future work help to develop a theoretical framework of how gesture is used to support learning in computing.
4.6 Conclusion

Chapters 3 and 4 explored the role of embodiment and developmental learning theory in computing education for young children aged six-to-seven years by presenting an in-depth analysis of RQ1 as to what type of interface (i.e., TUI or GUI) is the most beneficial for children in terms of learning outcomes, attitudes, enjoyment and gesture frequency and the effect of gender of the above measures. In contrast to Study 1, in Study 2 no difference was found between the TUI-PR and the GUI-PR in learning gains. However, the GUI-SR was associated with a greater improvement in children’s attitudes towards computing. This may be because, with the GUI, participants spent more time interacting with the device rather than waiting for the system to be set up by the researcher. Although the children in this study spent more interaction time with the GUI-SR and the TUI-PR session times were longer, both the TUI-PR and the GUI-SR were enjoyable, and the children produced gestures while using them. These results may save schools from the challenges of deploying TUI-PR, which cost more compared to GUI-SR (Sapounidis and Demetriadis 2017). There were no differences found between the above measures and gender.

The role of spontaneous gestures and their types that children use when solving programming tasks were explored. A video analysis scheme adapted from mathematics education research was used to code the spontaneous gestures produced during the learning sessions. In Study 2 a statistically significant relationship was not observed between gesture frequency and learning outcomes, which contested the finding of Study 1. It might be because the number of researcher gestures was not controlled between the groups and the most complex activity was removed from study 2. This hypothesis needs further investigation, especially in order to understand the role of gesture in computing education, by examining spontaneous gestures used in explaining complex programming activity with various age groups. This research contributes to the existing literature exploring children’s programming strategies and approaches, and to the current conversation about the role of gestures in computing.
The qualitative analysis of conceptual metaphor indicated that children use spontaneous hand gestures to demonstrate abstract computational concepts such as looping, which is examined further in the next study. Metaphors are important in human communication and reflect human cognition; people often use them to visualize their thinking (Martha W Alibali et al. 1999). Students and teachers routinely produce gestures to communicate information alongside speech and enhance listeners’ comprehension (Martha W. Alibali and Nathan 2012). Additionally, a growing body of evidence suggests that gesture plays a vital – potentially fundamental – role in knowledge development and change. Thus, understanding the use of gestures in classrooms is important in developing a deeper understanding of instructional communication and knowledge change (Martha W. Alibali and Nathan 2012). Consequently, Study 3 in the next chapter presents empirical findings for the embodied natural if computing concepts by analysing gestures generated by children aged six-to-seven years explaining basic programming concepts. The findings show representational patterns from these gestures, thereby suggesting the potential of this methodological approach to provide a deeper understanding of the nature of thinking in computer education.
CHAPTER 5
STUDY 3\textsuperscript{2}: INVESTIGATING PRIMARY SCHOOL CHILDREN’S EMBODIED EXPRESSION OF PROGRAMMING CONCEPTS

5.1 INTRODUCTION

Chapter 2 reviewed a body of literature that has emerged across various domains, including psychology, cognitive science, the learning sciences and human-computer interaction, offering evidence for the importance of understanding the role of gestures in learning (Sections 2.6). Chapter 3 (Sections 3.4.3.4) and Chapter 4 (Section 4.4.3.4) reported on children’s spontaneous use of representational metaphorical gestures (i.e., conceptual metaphors) when solving programming tasks. Notably, the use of conceptual metaphors may indicate that computing relies on embodied cognition which could reveal implicit knowledge and serve as a promising tool to reflect students’ understanding.

This chapter presents Study 3, which investigates the use of representational metaphorical gestures (i.e., conceptual metaphors) in primary school children aged between six and seven. The study seeks to apply a novel approach to help to reveal the conceptual foundations of basic programming. Specifically, the approach explores the role of conceptual metaphor via an analysis of children’s use of spontaneous gestures when explaining programming concepts. The study contributed to answering the third main research question of this thesis, namely:

**RQ3:** How are conceptual metaphors used by young children to explain programming concepts?

\textsuperscript{2} Submitted as ‘Investigating primary school children’s embodied expression of programming concepts’ to the International Journal of Child-computer Interaction.
• **RQ3.1**: What types of spontaneous gesture do young children use to explain programming concepts after an introductory programming activity?

• **RQ3.2**: What types of conceptual metaphor do young children use to explain programming concepts after an introductory programming activity?

• **RQ3.3**: Are there any differences evident in the use of gestures and conceptual metaphors when children are interviewed again two weeks after the introductory programming activity and first interview?

• **RQ3.4**: What is the effect of interface type (GUI versus TUI) previously used on the use of spontaneous gestures when explaining programming concepts?

The findings indicate that the participants who used metaphorical gestures drew on overarching embodied metaphors in their explanations. This provides evidence of the embodied nature of their computing concepts, which may drive future research into embodied cognition in the field of computing education. This research is a first step toward developing our understanding of young children’s embodied descriptions of programming concepts, as well as identifying conceptual metaphors that may support their learning.

### 5.2 Method

This study draws upon the theoretical framework of embodied cognition from the learning sciences, the methodological tools of cognitive linguistics and gesture research to investigate participants’ use of gesture when responding to interview questions. This approach builds on the work of Solomon et al. and Manches et al. who investigated the use of gesture in computing education with older learners (Manches et al. 2020; A. Solomon et al. 2018). This methodology was a good match for our aim of investigating the embodied representations (gestures) which children spontaneously produce while explaining computing concepts (Manches et al. 2020; A. Solomon et al. 2018).
The meaning of their gestures was interpreted from context, which was the verbal response to questions in a structured interview (Parrill and Sweetser 2005). Interviews are often used with children to prompt explanations of their understanding of computing (Mertala 2019; Robertson, Manches, and Pain 2017; Sheehan 2003).

5.2.1 Participants

The study was conducted after Study 2 with the same participants and at the same international school in Saudi Arabia where the children spoke Arabic and English alternately. Participants were from five single-gender classes (two girls’ classes and three boys’ classes) and all had participated in the previous study with the researcher. The children had learned fundamental computer concepts using block-based programming language with either a TUI-PR (a tangible user interface that controlled a physical robot) or a GUI-SR (a graphical user interface that controlled an on-screen robot). Thus, participants were familiar with the basic programming concepts asked about in the interview. 50 students participated in total (25 girls and 25 boys, all aged between six and seven). Each participant was interviewed twice (Interview 1 and Interview 2), with a two-week gap between the interviews. The first of the interviews took place immediately after the pairs had completed Study 2. Parents gave informed consent for their children to participate in the study. The focus on embodied responses in the study was not explicitly mentioned to participants, however, the parental consent form stated that this was the main focus of this study. Children gave verbal assent before each interview.

5.2.2 Data Collection

This study was conducted via interviews in which participants were asked to explain their understanding of computing concepts. The interview provided a context for the generation of rich data over a short period and consistently among participants.

The manner in which the questions (see Table 5.1) were asked influenced children’s explanations and thus the questions were intended to be as open and non-prescriptive as possible (Mertala 2019). The questions were asked first in English and then translated immediately into. The researcher used the two languages alternately as the children’s
teachers do in the classroom. Interviews were video recorded using a GoPro video camera positioned in such a way as to observe children’s hand movements.

In both interviews, all participants were asked to explain the same concepts in the same order. For some questions, participants were given a code snippet on a sheet of paper that they were then asked to explain. This provision may have influenced gestures, such as deictic gestures used directly toward the resources (M. Wilson 2002). Therefore, we recorded all the types of gestures, including the deictic gesturing.

This study adhered to the University of Sussex ethical guidelines and the Ministry of Education rules in Saudi Arabia. It was conducted at an international school in Riyadh, Saudi Arabia. Ethical approval was granted by both the University’s and the Ministry’s ethics committee.

5.2.3 Immediate and Delayed Interviews

This study was conducted via two interviews, the first interview aimed to capture students gestures immediately after the learning section. The second interview was two weeks after the introductory programming activity and first interview. Gestures can support learning over time, such as in mathematics (Martha Wagner Alibali and Goldinmeadow 1993; Cook and Goldin-Meadow 2006; Cook, Mitchell, and Goldin-Meadow 2008) and in learning a second language (Allen 1995) potentially be related to the role of gestures in encoding and retrieving information to and from memory (Cook, Yip, and Goldin-Meadow 2010). Often, studies investigating the use of gestures for the immediate impact, but few studies had looked at the long-term impact. Previous research on gestures has shown that the effect of engaging in gesturing was higher on the long-term assessment (over three weeks) compared to the short term (Cook, Mitchell, and Goldin-Meadow 2008a; Cook, Yip, and Goldin-Meadow 2010a) or even two days after learning (Hornstein and Mulligan 2004). This might be related to the learning effects after sleep on memory. During sleep, learning-related neural networks are reactivated (M. A. Wilson and McNaughton 1994). The periods between the first and second assessment in the literature varied from two days to three weeks; therefore, in this experiment, the second
interview was conducted two weeks later due to the school schedule. Also, two weeks will give children time to think about the topic and reduce (if any) the influence of researcher gestures on the learning session. No prior research has analysed the development of gestures and whether changes occur in terms of how abstract concepts are represented over time, and so this represents the focus of the current research.

5.2.4 Interview Questions

The first question, (Q1) was an ice breaker for the interview session to help make participants more comfortable. It was added to the script after the first group was interviewed because the researcher noticed that participants seemed shy about the interview setting. Subsequent questions prompted the children to explain the meaning of ‘program’ (Q2), the code of a simple program (Q3), the meaning of ‘iteration’ (Q4) and the code of a complex program including iteration (Q5). We selected the concepts ‘program’ and ‘iteration’ because they are age-appropriate foundational concepts that primary children need to understand and use (Berry 2015; Futschek and Moschitz 2011; Robertson, Manches, and Pain 2017). Iteration is considered a complex concept for young children that may require them to offload their cognitive work in explaining it and use an embodied representation (Futschek and Moschitz 2011).

Table 5.1 Interview questions

<table>
<thead>
<tr>
<th>Number</th>
<th>Question</th>
<th>Themes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>What did you learn today/last time in our session with the robot?</td>
<td>Icebreaker</td>
</tr>
<tr>
<td>Q2</td>
<td>What is a program?</td>
<td>Explaining the general concept of a program</td>
</tr>
<tr>
<td>Q3</td>
<td>Explain this program, please? [showing the sequence program; see Figure 5.1(a)]</td>
<td>Describing how a simple program works using simple sequence commands</td>
</tr>
<tr>
<td>Q4</td>
<td>What does repeat do? [If the child did not answer, the researcher would show the child an image of the Repeat block; see Figure 5.1(b)]</td>
<td>Explaining complex computing concepts such as iteration</td>
</tr>
<tr>
<td>Q5</td>
<td>Explain this program, please? [showing the complex program; see Figure 5.1(c)]</td>
<td>Describing how a complex program works</td>
</tr>
</tbody>
</table>
5.2.5 Procedure and Materials

This study involved the same participants as Study 2. The first interview (Interview 1) of Study 3 began immediately following the completion of Study 2. Interview 1 aimed to capture the immediate representation of the programming concepts after the programming session. The author interviewed the participants individually in a quiet room. The second interview (Interview 2) was held two weeks later, the setting for which was also a quiet room. This approach sought to give children time to grasp the knowledge they had learned in Study 2. It also enabled the investigation of RQ3.3 and RQ3.4 to determine whether the participants' use of conceptual metaphors changed over time; and it assessed the effect of time on the use of gestures when explaining programming concepts. The use of representations in the material can increase children’s familiarity. Children may draw on these representations from cultural knowledge (e.g., road arrows indicating direction).

The same questions were asked in both interviews, with the only change being to Q1, which changed from ‘What did you learn today in our session with the robot?’ to ‘What did you learn last time in our session with the robot?’ Participants were asked to stand when they were interviewed, as Manches et al in 2019 reported that sitting may obscure gesturing (Manches et al. 2020). The interviewer then asked the questions in Table 5.1. Each interview took around two-to-three minutes and the participants were all given the questions in the same order. During the interviews, prompts were given to clarify what participants meant or to correct their code reading (see Table 5.2).

The participants answered the questions using Arabic and English interchangeably, often switching between the languages in a single sentence. Therefore, the language that each participant used was not coded.

The interviewer thanked the participants every time they answered a question. The interviewer tried to avoid making representational gestures because this may have led to mirroring. After the participants had finished, the interviewer thanked them again for their participation.
In contrast to Study 1 and Study 2, the researcher gestures were not recorded and quantified. The aim of study 3 was to observe children’s use of gestures when explaining programming concepts. The children were familiar with the programming concepts as they completed Study 2, which meant the researcher/ interviewer did not have to explain any programming concepts or use any gestures when she asked the questions. The researcher did not gesture to ensure that no undue influence was exerted on the children’s gestures which is an important point because previous studies have shown that teachers’ gestures may influence students. Noteworthily, this was not the case in studies 1 and 2.

![Figure 5.1](image)

(a) Q3 - Simple code aimed to describing how a simple program works; (b) Repeat block; (c) Q5 - Complex code aimed to describing how a complex program works

5.2.6 Measures and Data Analysis

The researcher recorded and processed the videos of participants answering the interview questions. She then prepared them for analysis, which included reducing the data into themes through via a coding scheme, data display and conclusion (Miles and Huberman 1984). Each recording was first reviewed, then translated from Arabic to English if needed and, finally, transcribed. A structured Microsoft Excel spreadsheet was created with the following categories for each question: researcher talk, gestures, participant’s answer, participant’s gesture type and gesture description. The researcher filled in the categories in which she believed the researcher or participant had responded verbally or used gestures.
while asking and answering the questions. Additionally, the researcher removed irrelevant parts of the interaction that were not related to the questions.

The researcher coded all the video data. A random sample of 20% of the data was generated for second coding by the main supervisor. Inter-rater reliability was determined using Cohen’s Kappa scores, with substantial agreement based on Fleiss (Fleiss, Levin, and Paik 2004). Results were as follows: Kappa score for the gesture type was 0.681.

**VERBAL RESPONSES**

The verbal responses for the questions were transcribed and used to interpret the meaning of children’s use of gestures, but not otherwise analysed as part of this study.

**GESTURE RESPONSES**

The author categorised and coded all the gestures, drawing upon the children’s speech to guide the interpretation and recorded them in the transcript and the Excel spreadsheet. The gesture responses were then categorised based on Alibali and Nathan’s scheme (2012) and divided into two types of gesture, as follows:

- **Deictic gesturing (e.g., pointing):** reflected the grounding of cognitive gestures in the physical environment (e.g., pointing to the function block).
- **Representational gesturing (RG):** including gesturing that represented concrete or abstract concepts either by hand (hand shape, or motion trajectory) or by body (e.g., rotating the whole body to indicate a turn). It could be either a:
  - Literal gesture (i.e., iconic gesture): depicted aspects of meaning. It is important to note that in our analysis when a participant pointed their index finger to the right while explaining the Right block concept, we considered this to be an iconic gesture not referring to an object in the environment, or a
  - Metaphorical gesture: e.g., identifying how children represented the abstract notion of iteration.

For the metaphorical gestures identified, the author and the main supervisors worked together to identify conceptual mapping patterns across participants. Conceptual mapping
was the process of mapping between the gesture and conceptual entities in the children’s speech.

5.3 RESULTS

This section presented the results that contribute to answering RQ3 ‘How are conceptual metaphors, as externalised through gestures, used by young children to explain programming concepts?’ (Section 5.3.1 and Section 5.3.2).

5.3.1 Overview of the Questions’ Responses

The number of questions asked, the responses and prompts for each question as well as the prompt types are shown in Table 5.2 (e.g., in Interview 1, Question 1 was asked 45 times to the 45 participants, received 38 responses and 10 gestures, no prompts were given). During the data collection and due to technical issues with the camera, we lost nine recordings (three in Interview 1 and five in Interview 2) and one participant withdrew from Interview 2. Additionally, due to interviewer error, not all participants were asked Q2.

Identifying the prompt type was vital to make sure that participants’ gestures were not influenced by any researcher prompts. More prompts were given in the first interview than the second because participants were not familiar with the interview setting and the questions. The number of prompts was higher in questions in which participants were asked to explain written code (Q3, Q5). Participants sometimes just read the code, so the researcher prompted them to explain the code instead of just reading.
Table 5.2 Prompts and responses for each question

<table>
<thead>
<tr>
<th>Q#</th>
<th>Interview</th>
<th>Asked</th>
<th>Answers Before prompted</th>
<th>Gestures Before prompted</th>
<th>Prompted</th>
<th>Prompt types</th>
<th>No answer at first</th>
<th>The total of answers After prompt</th>
<th>The total Gestures After prompt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>45</td>
<td>38</td>
<td>10</td>
<td>0</td>
<td></td>
<td>0</td>
<td>38</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>45</td>
<td>43</td>
<td>5</td>
<td>0</td>
<td></td>
<td>2</td>
<td>43</td>
<td>5</td>
</tr>
<tr>
<td>Q2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>45</td>
<td>34</td>
<td>10</td>
<td>0</td>
<td></td>
<td>11</td>
<td>45</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>37</td>
<td>22</td>
<td>5</td>
<td>0</td>
<td></td>
<td>15</td>
<td>22</td>
<td>3</td>
</tr>
<tr>
<td>Q3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>47</td>
<td>47</td>
<td>186</td>
<td>105</td>
<td></td>
<td>0</td>
<td>47</td>
<td>261</td>
</tr>
</tbody>
</table>
|    | 2         | 45    | 45                     | 260                     | 60      | ▪ Asking the student to demonstrate the action (e.g., ‘Show me how’, or ‘How?’)  
▪ Indicating the start point of the code (‘We start from here’.)  
▪ ‘Left is this way [pointing to the left]’.)  
▪ Encouraging them to explain the whole code sequence using ‘then’ or ‘continue’. | 0                 | 45                                | 304                             |
<table>
<thead>
<tr>
<th></th>
<th>Q4</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>Q5</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>47</td>
<td>40</td>
<td>51</td>
<td>7</td>
<td>Showing the image of</td>
<td>3</td>
<td>44</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>35</td>
<td>33</td>
<td>60</td>
<td>10</td>
<td>the repeat</td>
<td>5</td>
<td>41</td>
<td>68</td>
</tr>
<tr>
<td></td>
<td>Q5</td>
<td>1</td>
<td>47</td>
<td>47</td>
<td>137</td>
<td>22</td>
<td>3</td>
<td>47</td>
<td>151</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>45</td>
<td>44</td>
<td>188</td>
<td>22</td>
<td>Same as Q3</td>
<td>1</td>
<td>45</td>
<td>202</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>1</td>
<td>231</td>
<td>206</td>
<td>394</td>
<td>134</td>
<td>14</td>
<td>211</td>
<td>484</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>218</td>
<td>186</td>
<td>518</td>
<td>92</td>
<td>23</td>
<td>37</td>
<td>566</td>
<td>1068</td>
</tr>
<tr>
<td></td>
<td>Overall total</td>
<td>1 and 2</td>
<td>449</td>
<td>392</td>
<td>899</td>
<td>226</td>
<td>37</td>
<td>566</td>
<td>1068</td>
</tr>
</tbody>
</table>
5.3.2 Types of Spontaneous Gesture when Explaining Programming Concepts

This section presents a general analysis of the types of spontaneous gesture used to explain programming concepts in the two interviews. This section addresses RQ3.1: ‘What types of spontaneous gestures do young children use to explain programming concepts after an introductory programming activity?’ In total, the author coded 1,068 gestures into three types: pointing and RG either as a literal or as a metaphor. See Figure 5.2 below for examples for each type.

![Figure 5.2 Gesture type illustrations (a) Participant pointing to the code; (b) Participant using RG-literal to describe the left turn; (c) Participant moving their hand in a circular motion to represent repeat.](image)

The 1,068 gestures were coded across 566 explanations (answers to five questions from Interview 1 from 45 participants and Interview 2 from 47 participants). Three participants generated no gestures. The total number of participants’ gestures in Interview 1 ranged from 0 to 26, with a median of 11, while that for Interview 2 ranged from 0 to 21 (median 13).

Deictic Gestures

A deictic gesture (e.g., pointing) was used to indicate an object’s location (McNeill 1994). Participants pointed frequently when physical material was provided, such as in Q3 and Q5, by indicating a code block while explaining its functionality. These gestures guided both the participants and researcher to the related block. Tracing the code might have helped the participants while they tried to understand the control flow of the code. Deictic gestures may have reflected cognition and revealed students’ understanding of the code execution, similar to Solomon et al.’s 2018 findings (A. Solomon et al. 2018) . For
example, when participants pointed to the left block in a simple program and said ‘right’ whilst gesturing to the left, this let the researcher know that the participant had issues reading and/or understanding the code or just muddled right and left.

**LITERAL GESTURES**

A literal representational gesture (RG-literal) represents a concrete idea and conveys information about the size, shape or orientation of a discourse object (Hostetter and Alibali 2008). Such gestures are used to facilitate communication and simulate an action. Participants used literal gestures mostly to simulate the robot’s actions – forward, right and left. Some participants used literal gestures to represent a number that was important in the code. For example, one participant said ‘the robot would repeat the code twice’ with a hand gesture for the number, two.

**METAPHORICAL GESTURES**

Metaphorical representational gestures (RG-metaphorical) represent abstract ideas (Roth 2001). An abstract noun, as defined by Lexico, quoting from the Oxford Dictionary (2020), is ‘a noun which refers to ideas, qualities and conditions - things that cannot be seen or touched and things which have no physical reality’ (Lexicon 2018). Therefore, we considered the concepts of program and iteration as abstract concepts, but we also noted that the noise block was a unique case. It is not an abstract noun, but it is not concrete visually. Therefore, we considered the gesture that represented the noise and repeat block to be RG-metaphorical. Participants produced a total of 53 RG-metaphorical gestures (Interview 1 = 24; Interview 2 = 29).

5.3.3 **Types of Conceptual Metaphor Used to Explain Programming Concepts**

This section explores the metaphorical gestures and what they reveal about how children conceptualised computing concept in order to answer to RQ3.2: ‘What types of conceptual metaphor do young children use to explain programming concepts?’

**NOISE BLOCK**

The noise block, which makes a buzzing sound and does not involve any movement of the robot, was part of the simple program that participants explained.
A noise block is a common programming block used in robot programming block environments, beyond movement-related blocks. The noise block is one of the few blocks that were not tied to spatial elements of the programming task. Therefore, the noise block cannot be represented explicitly as concrete representation; it is also potentially more likely to be related to metaphorical representation.

A few participants demonstrated the noise with a gesture and a few said the word, ‘pop’. Over both interviews, 15 RG-metaphorical gestures and 30 pointing gestures were generated for the noise block and there were 45 responses where no gesture was produced when the noise block was reached in Q3.

Participants represented the noise metaphorically in many different ways, either by keeping the same position, representing the freeze by an action, or by demonstrating a movement. First, a few participants represented the noise as a moment of freezing, where the robot stopped moving and kept its previous position, which is exactly what the robot did, e.g., P40. Others represented the freezing by an action. For example, P26 and P27 clenched their hands, whereas P50 clasped her hands. P28 posed his hands as if they were holding something, whereas P18 held a handout flat and said, ‘pop,’ while P10 hit the table with her fist as a gesture to represent the noise. Other participants represented the noise block with different movements, even though the robot in the learning session did not perform any movement when it produced the noise. For example, P23, P40 and P42 used their whole body to dance when they reached the noise block, whereas others just moved their hands. P31 moved his hand left and right, whereas P35 moved his hands toward the front. P09 moved both of her hands in a circular motion as if they were a hurricane. P45 moved his hands in an arc motion.

**PROGRAM**

Only two participants used gestures when asked about the meaning of a program and they both answered that the program means repeat. For example, P37 said that the program was a ‘repeat’ while gesturing the form of a container. Therefore, their gestures were coded as a metaphorical representation of the repeat.
REPEAT

In Interview 1, 18 gestures represented the repeat either as a circle (8), an arc (6), a forward line (2), or a flowing arrow mark on the repeat block (2). In Interview 2, 20 gestures represented the repeat either as a circle (8), a container (4), an arc (4), through repetitive movements such as a wave (1), or by drawing the arrow found on the repeat block (1) and performing the action (2).

COMPUTING CONSTRUCTS AS PHYSICAL OBJECTS

When explaining computing concepts, two participants simulated a pinching action described by Edwards in a 2009 study focusing on mathematical concepts (Edwards 2009a). Similar to the concept of numbers that can be represented physically through tangible manipulatives and how children draw on these physical representations when thinking about numbers, participants’ experience with programming blocks might influence them to draw upon experiences of manipulating tangible representations of computing concepts. A pinch was used when describing programming blocks inside the repeat (e.g., see Figure 5.3(a)). Additionally, participants represented repeat as a container (see Figure 5.3(b)), by creating a boundary of an object and using another hand to gesture into the hand. This was often associated with indicating that the code inside the repeat will be executed.

COMPUTING PROCESSES AS A MOTION ALONG A PATH

There are three main axes related to the body: longitudinal, frontal (see Figure 5.3) and transverse (see Figure 5.4):

LONGITUDINAL AXIS: the path up-to-down in front of the body, paralleling how the participants wrote the code from top to bottom. For example, a gesture referring to time-based computing processes might go downward in a longitudinal direction. One participant (P41) produced this gesture and said, ‘Repeat’ while his arm moved downward. He discussed algorithmic steps/instructions, while making a step going downward with his hand see Figure 5.3(c).
**FRONTAL AXIS:** this is the pathway moving away from the body where a circular gesture is projected outward from the body. Two participants (Interview 1: P44, Interview 2, P05) represented the repetition with two straight lines going forth and back (such as P44, see Figure 5.3(d)). This supports the idea that participants were drawing a general body-based spatial metaphor of time and that the future was in front of them, conceptualising the robot movement.

**TRANSVERSAL AXIS:** this is the pathway moving from left to right across the body, or vice versa. English is written from left-to-right, whereas Arabic, the participants’ native language, is written from right-to-left. This suggests that gestures communicating computational processes might trace a similar transversal axis. Twelve participants drew the arc/circular shape from right to left, anticlockwise (six participants in Interview 1 and six participants in Interview 2; see Figure 5.4). Gesture movement pathways were noticed when participants talked about repeating a program or what they had just said. For example, P20 read the first line of the code for Q3, ‘left’, then said, ‘It will do it again’, with her right hand (RH) moving in an arching gesture. Another gesture represented the repetition as a wave or climbing a hill from left-to-right (see Figure 5.6(a)). Moreover, some participants represented one circle of the repeat clockwise and the second circle anticlockwise (see Figure 5.6(c)).

**GESTURE SIZE AND FREQUENCY**

The shape of the hand and whether one or two hands were used suggested differences in the perspective of the size of an imaginary object. Participants had different perspectives of the ‘size’ because, in CS, physical size has no meaning. For example, P09 represented the repeat using one big circle in both interviews, whereas P19 represented the repeat with a small circle. P36 represented the repeat with two circles, where the first was bigger than the second (see Figure 5.6(c)). Additionally, the circular movement frequency was not consistent across participants, ranging from one to three cycles. Figure 5.4 to Figure 5.6 illustrate the different representations and frequencies of the circular motion.
P10: ‘We used that [referring to repeat] that if we put two things will go and will return again.’

P17: ‘Put repeat [hand gesture as container], then the left, forward, right [2 RG-literal] and the sound.’

P41 moves his hand downward and says, ‘Repeat’.

P44’s hand moves two times forward

(a) (b) (c) (d)

Figure 5.3 Motion gesture examples (a) Pinch; (b) Container (c) Longitudinal movement; (d) Frontal movement.

P09 gesture: ‘It repeats. If you did it wrong, it would repeat it again.’; with her palm open.

P37: ‘Do it again’. He drew two circles on the table with his hand closed.

P39 pointed to the code from the previous question and said ‘It does it double times’, gesturing with both hands in a circular motion.

(a) (b) (c)

Figure 5.4 Transversal axis examples (a) One big circle, clockwise; (b) Drawing two circles clockwise on the table; (c) Moving both hands at the same time in an anticlockwise motion, three times.
P19: “If we do a thing, we will do it again”, making a small circular motion with her hands clasped.

P32: 'You can put things like left, then it will move left left, then it will go to the flag. The repeat will make it go again' [hand gesture in an arc]

P18: 'Here is repeat' pointing to the repeat. 'Inside there is right, right' [pointing inside the repeat]. 'It will repeat again and again' [big circular motion with her hand open].

Figure 5.5 Transversal axis examples (a) Grasping with a small circle (clockwise); (b) Arc (clockwise); (c) Two big circles (anticlockwise).
P19: 'I there is something here [a], it will move twice [b].'

P01: 'Doing it again and again [hand moving from right to left] and again'.

P36: 'Do it again! He drew a big circle on the table with his hands gripped clockwise, followed by a smaller circle and hand-moving when talking about repeat anticlockwise.

(a) (b) (c)

Figure 5.6 Transversal axis examples (a) Waves (anticlockwise); (b) Right-to-left movement (anticlockwise); (c) Two circles (clockwise and anticlockwise).
In two instances, when participants answered the question, ‘What is a program?’, their use of a gesture revealed implicit knowledge which they did not express in words. For example, P45 said: ‘Yes program the stuff, like the robot’. It was not clear what participants meant when they said ‘stuff’; it was ambiguous. However, their gesture, hands moved on top of each other, represented the order of the programming blocks used in the learning activity (see Figure 5.7). In another example, P07 answered the question, ‘What is a program?’ by responding, ‘Do something now.’, while holding both hands together. The participant then continued, ‘Then you do the same thing there.’, while turning their body to the side and turning their hands. The researcher did not know what the student meant by ‘something’ and ‘the same thing’. From the participant’s gestures the researcher understood ‘something’ as meaning the robot’s movements. The participant was simulating its movement by moving their body.

Figure 5.7 P45 use of gesture to reveal implicit knowledge

5.3.4  Representational Gesture Changes Over Time

This section describes metaphorical gesture changes across the two interviews to answer RQ3.3: ‘Are there any differences evident in the use of gestures and conceptual metaphors when children are interviewed again, two weeks after the introductory programming
activity, compared with that in a first interview immediately after the activity?’. The process compared the use of conceptual metaphors in each interview.

- Agreement-level 1: The concept was represented twice in one interview and followed the same category (e.g., P10).
- Agreement-level 2: The concept was represented in the same category in Interviews 1 and 2 (e.g., P09).
- Disagreement-level 1: The concept was represented twice in one interview and each time was in different category than the other (e.g., P19).
- Disagreement-level 2 The concept was represented differently in each interview (e.g., P18).
- Solo: The concept was represented once in Interview 1 or 2 (e.g., P01).

For the noise block, one participant generated an RG-metaphorical gesture in both interviews with consistent representations. Interview 1 featured five representations while Interview 2 featured eight (see Table 5.3). For the repeat concept, three participants (P09, P10, P20) maintained Agreement-level 2 for the repeat representation, while 10 participants represented the repeat differently in each interview (Disagreement-Level 2). The other six represented the repeat solo in Interview 1 or Interview 2. Additionally, a more detailed breakdown is shown for repeating concepts twice in the same interview (five disagreements and one agreement).

Table 5.4 illustrates the changes in the metaphorical representation.
<table>
<thead>
<tr>
<th>P</th>
<th>Interfaces</th>
<th>Interview 1</th>
<th>Interview 2</th>
<th>Compatibility</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Qs</td>
<td>Gesture description</td>
<td>Gesture category</td>
</tr>
<tr>
<td>P09</td>
<td>TUI-PR</td>
<td>Q3</td>
<td>Moving both had in circular motion as if it is a hurricane</td>
<td></td>
</tr>
<tr>
<td>P10</td>
<td>TUI-PR</td>
<td>Q3</td>
<td>Hit the table with her fist and said ‘Make a sound’</td>
<td>Freeze by action</td>
</tr>
<tr>
<td>P18</td>
<td>GUI-SR</td>
<td>Q3</td>
<td>Held both hand and said ‘pop’</td>
<td>Freeze by action</td>
</tr>
<tr>
<td>P23</td>
<td>TUI-PR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P26</td>
<td>TUI-PR</td>
<td>Q3</td>
<td>Hand as a grip</td>
<td>Freeze by action</td>
</tr>
<tr>
<td>P27</td>
<td>TUI-PR</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5.3 Summary of noise representational gesture
<table>
<thead>
<tr>
<th>P28</th>
<th>GUI-SR</th>
<th>Q3</th>
<th>Hands as if they are holding something</th>
<th>Freeze by action</th>
<th>Solo</th>
</tr>
</thead>
<tbody>
<tr>
<td>P31</td>
<td>GUI-SR</td>
<td>Q3</td>
<td>Hand moving right to left</td>
<td>Move</td>
<td>Solo</td>
</tr>
<tr>
<td>P35</td>
<td>TUI-PR</td>
<td>Q3</td>
<td>Hand movement to the front</td>
<td>Move</td>
<td>Solo</td>
</tr>
<tr>
<td>P40</td>
<td>TUI-PR</td>
<td>Q3</td>
<td>Handstand in the same previous position</td>
<td>Freeze</td>
<td>Solo</td>
</tr>
<tr>
<td>P42</td>
<td>GUI-SR</td>
<td>Q3</td>
<td>Dance</td>
<td>Move</td>
<td>Solo</td>
</tr>
<tr>
<td>P45</td>
<td>GUI-SR</td>
<td>Q3</td>
<td>Circular motion</td>
<td>Move</td>
<td>Solo</td>
</tr>
<tr>
<td>P49</td>
<td>GUI-SR</td>
<td>Q3</td>
<td>Acting the noise</td>
<td>Move</td>
<td>Solo</td>
</tr>
<tr>
<td>P50</td>
<td>GUI-SR</td>
<td>Q3</td>
<td>Clasped her hands</td>
<td>Freeze by action</td>
<td>Solo</td>
</tr>
<tr>
<td>Participants</td>
<td>interfaces type</td>
<td>Q</td>
<td>Gesture description</td>
<td>Gesture category</td>
<td>Q</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------</td>
<td>---</td>
<td>-------------------------------------------------------------------------------------</td>
<td>------------------</td>
<td>---</td>
</tr>
<tr>
<td>P01</td>
<td>GUI-SR</td>
<td>Q4</td>
<td>One, big circle gesture with hand palm open, on the air, moving from right to left</td>
<td>Transversal axis - anticlockwise</td>
<td>Q4</td>
</tr>
<tr>
<td>P05</td>
<td>TUI-PR</td>
<td>Q4</td>
<td>Gesture with the word ‘again’ hand on the table and moving forward twice</td>
<td>Frontal</td>
<td></td>
</tr>
<tr>
<td>P09</td>
<td>TUI-PR</td>
<td>Q4</td>
<td>One, big circular movement with finger as number 2, on the air</td>
<td>Transversal axis - Clockwise</td>
<td>Q4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>----</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>139</td>
<td></td>
<td>In arc shape from right to left the hand goes from one position to another</td>
<td>Transversal axis - anticlockwise</td>
<td></td>
<td>Disagreement-level 1</td>
</tr>
<tr>
<td>P10</td>
<td>TUI-PR</td>
<td>Q4</td>
<td>Hand on the repeat arrow, with circular motion,</td>
<td>Transversal axis - Clockwise</td>
<td>Q4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q5</td>
<td>Hand movement following the repeat arrow</td>
<td>Transversal axis - Clockwise</td>
<td></td>
</tr>
<tr>
<td>P17</td>
<td>TUI-PR</td>
<td></td>
<td></td>
<td>Iconic gesture represents the shape of the repeat</td>
<td>Q4</td>
</tr>
<tr>
<td>P18</td>
<td>GUI-SR</td>
<td>Q4</td>
<td>2 big circle movement with pointy finger, on the table</td>
<td>Transversal axis - anticlockwise</td>
<td>One big circular motion in the air</td>
</tr>
<tr>
<td>P19</td>
<td>TUI-PR</td>
<td>Q3</td>
<td>Wave, left to right</td>
<td>Transversal axis - Clockwise</td>
<td>Disagreement - level 1</td>
</tr>
<tr>
<td>-----</td>
<td>--------</td>
<td>-----</td>
<td>----------------------</td>
<td>-----------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Grasping the hands, one small circular motion, right to left</td>
<td>Transversal axis - anticlockwise</td>
<td></td>
</tr>
<tr>
<td>P20</td>
<td></td>
<td>Q2</td>
<td>Arc</td>
<td>Transversal axis - Clockwise</td>
<td>Disagreement - level 1</td>
</tr>
<tr>
<td>P20</td>
<td></td>
<td>Q3</td>
<td>‘Then will do it again’ pointy finger moves in arc shape</td>
<td>Transversal axis - anticlockwise</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q4</td>
<td>One small circular motion on the table/ from right to left</td>
<td>Transversal axis - anticlockwise</td>
<td>Agreement - level 2</td>
</tr>
<tr>
<td>P28</td>
<td>GUI-SR</td>
<td>Q4</td>
<td>A container</td>
<td></td>
<td>Solo</td>
</tr>
<tr>
<td>P32</td>
<td></td>
<td>Q4</td>
<td>Hand in arch shape Transversal - Clockwise</td>
<td>Missing</td>
<td>Solo</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>P36</td>
<td>Q4</td>
<td>Drawing one big circle then small circle in the table with hand gripped, then movement front, from left to right Transversal Clockwise</td>
<td>Missing</td>
<td>Solo</td>
<td></td>
</tr>
<tr>
<td>P37</td>
<td>TUI-PR</td>
<td>Q2</td>
<td>As a container</td>
<td>Disagreement-level 1</td>
<td></td>
</tr>
<tr>
<td>P39</td>
<td>GUI-SR</td>
<td>Q4</td>
<td>Two fingers indicated number 2 and moving in a circular motion 3 times on the air Transversal axis - Clockwise</td>
<td>Q2</td>
<td>both hands toward each other and moving in circular motion three times Transversal axis - anticlockwise</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>P40</td>
<td>TUI-PR</td>
<td>Q4</td>
<td>Arm in a circular motion</td>
<td>Transversal axis - Clockwise</td>
<td>Q4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q5</td>
<td>Circular motion with pointy finger</td>
<td>Transversal axis - Clockwise</td>
<td></td>
</tr>
<tr>
<td>P41</td>
<td>TUI-PR</td>
<td>Q4</td>
<td>Hands moving in front of the body TWO times</td>
<td>Frontal</td>
<td>Q4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q5</td>
<td>The Arm move downward</td>
<td>longitudinal</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P42</td>
<td>GUI-SR</td>
<td>Q4</td>
<td></td>
<td>Q4</td>
<td>Moving in half line with pointy finger to the front</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P44</td>
<td>GUI-SR</td>
<td>Q4</td>
<td>The two hands in afront of each other the pointy finger is showing and one hand moving 2 circler motion</td>
<td></td>
<td>Q4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Disagreement-level 2 and 1
<table>
<thead>
<tr>
<th>P47</th>
<th>GUI-SR</th>
<th>Q4</th>
<th>and the other hand follow one circler. Anticlockwise Transversal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Pointy finger drawing a one small circler, right to left</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Transversal axis - anticlockwise</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Repeated drawing the arrow that it in the repeat block</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Transversal axis - Clockwise</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Disagreement-level 2</td>
</tr>
<tr>
<td>P49</td>
<td>GUI-SR</td>
<td>Q4</td>
<td>Hand moving in a small arc</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Transversal axis - Clockwise</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Q4 As a container</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Disagreement-level 2</td>
</tr>
</tbody>
</table>
An independent-samples t-test was run to determine differences in the number of overall gestures between the two interviews. Participants in Interview 2 made more gestures (N=45, M= 12.98, SD = 5.4) than in Interview 1 (N=47, M=10.30, SD=5.7) and the difference was statistically significant (95% CI, -5.007 to -3.353), t (90) = -2.288, p =.024).

5.3.5 The Effect of The Interface Type on The Use of Gesture When Explaining Programming Concepts

This section presents the effects of the interface type participants had used in the learning session in Study 2 to answer RQ3.4: ‘What is the effect of interface type (GUI versus TUI) on the use of spontaneous gesture when explaining programming concepts?’ For the noise concept, six participants who generated RG-metaphorical gestures were from the TUI-PR group and seven in the GUI-SR group in Study 2 (see Table 5.3). The repeat concept involved eight representations from the GUI-SR group and nine representations from TUI-PR groups (see Table 5.4). An independent-samples t-test determined differences in the number of overall gestures between interface types in both interviews. The TUI-PR users had a slightly larger mean number of gestures (M= 12, SD = 5.2) than the GUI-SR users (M=11.36, SD=6.0), but the difference was not statistically significant (95% CI, -1.805 to 3.091), t (90) = .522, p =.603.

5.4 Discussion

Our overall goal in this study was to answer RQ3: ‘How are conceptual metaphors, as externalised through gestures, used by young children to explain programming concepts?’ (including RQ3.1 to RQ3.4), by recording and analysing the verbal and gesture responses that children used to describe computing topics after experiencing a programming activity, including the concepts of a program and of iteration. We explored how children used spontaneous gestures to convey computing concepts and how they conceptualised these concepts. Additionally, we developed an analytic framework suitable for understanding gestures generated within this domain. This study’s findings indicate that, even for abstract concepts in computing education, the embodied nature of participants’
understanding of computing concepts was made evident through their gestures. The participants also produced a high number of literal representational gestures that mostly referred to concrete objects or processes. These often related to tangible materials utilised in the programming blocks or to the robot’s movement.

5.4.1 RQ3.1 What Types of Spontaneous Gesture Do Young Children Use to Explain Programming Concepts?

The study identified 281 pointing, 732 RG-literal and 53 RG-metaphorical gestures. Our findings support one of the central claims of embodied cognition: that offline thinking involves mental simulations of perception and action. The representational gestures were produced in the absence of ‘relevant stimuli’, e.g., a physical object such as a screen or board to point to. Importantly, the representational nature of these gestures provided additional information to their speech to help us to analyse their thinking in this domain. Our findings suggest that embodied cognition has many insights to offer to computing education (Grover and Pea 2013) in which subjects would benefit from a greater understanding of conceptual development. This study shows that the children aged six-to-seven years old produced less gestural evidence of metaphorical understanding compared to that of conceptual understanding. Only 5% of the generated gestures were metaphorical, so spontaneous metaphorical thinking did not play a strong role in these children’s reasoning about CS concepts. This is not surprising as the use of such metaphors is sophisticated and occur more frequently with older children.

Another explanation might be that the participants who represented the repeat concept metaphorically understood it. In 2009, Bakker et al. showed that abstract concepts, after first being understood, can be represented in body movements before they can be explained in words. Therefore, the limited use of metaphorical gestures may reflect a limited understanding of programming concepts (e.g., iteration and program). If children experience difficulties in explaining the concept in words and gestures, it is an indication that the concept is difficult for the age group (Antle et al. 2009). Further work should look at the use of metaphors in CS with older children.
5.4.2 RQ3.2 What Types of Conceptual Metaphor Do Young Children Use to Explain Programming Concepts?

Despite the participants’ young age and having no previous experience with a programming language (except for the 25-minute learning session in Study 2), they generated 52 RG-metaphorical gestures. Several patterns of gesture were identified. Patterns included using a circular motion to communicate the repeat block with different frequencies and sizes and different hand grasps to represent the robot’s non-movement when making a noise.

Most of the RG-metaphorical gestures that were identified described either a repeat or a noise. The way participants gestured suggested their degree of understanding. It was not possible to test the accuracy of these postulated degrees of understanding independently, but with this caveat, we offer some plausible interpretations. For example, participants representing the noise block with a movement might have indicated a low understanding of the noise block’s functionality because the robot did not move when making a noise in the learning session. Conversely, a no-movement representation was either a literal representation of the freeze or metaphorical; either representation might indicate that participants understood that the noise block meant that the robot would stay in the same position. A literal representation of the freeze means simulating the robot’s movement when it runs the noise block by keeping the previous gestures that participants made for the previous action block and saying ‘noise’ or ‘sound’. In contrast, a metaphorical representation of the noise block as freeze involved the child using a gesture such as clenching, clasping or hitting the table to indicate his or her understanding that the freeze block caused the robot to remain stationary. This could indicate how students had understood the idea and could be used by teachers to glean information about their students’ learning stage.

Also, the frequency of circular gestures – ranging from one to three may indicate the repeated execution of a program (once, twice, or thrice) based on the number of times the child performed the circular motion. This could be reflective of the children’s understanding of the repeat concept. Noteworthy, teachers can use the information that
students convey with their hand movements to guide and assist their learning. As a case in point, Kelly et al. found that teachers can assess understanding by considering students’ gestures (Kelly et al. 2002). This was attributed to the fact that the information students convey using gestures is at the cutting edge of their knowledge (e.g., the noise block was not a moving action block, and they were aware of its functionality).

The approach used to categorize RG-metaphorical gestures in this work was similar to that of (Manches et al. 2020). In their approach, the authors categorised the gestures of undergraduate computer science students into the following image schemes: the computer as an object or container and the path-course goal schemes that serve as the foundation for mathematical concepts. It is worth noting that the metaphor outlined here does not differ between adults and children, but varying forms of metaphors arose. As a case in point, in contrast to the non-Arabic speakers in Manches et al.’s study, who all represented iteration using a clockwise gesture, half of the participants in this study represented the repeat in an anticlockwise circle. This may be due to the difference in meaning implied by the left-right/right-left text direction in English/Arabic. It is notable that half of our participants used a anticlockwise gesture despite the blocks-based language used in the learning activity representing the control flow of the repeat block in a clockwise direction. Insofar as the features of languages impact human conceptualisations of their surroundings, students’ native languages may influence their thought in a clear semantic way (Amici et al. 2019). To increase the accessibility of programming and computing education among learners who do not speak English (Amici et al. 2019; GOV.UK 2013; Manches et al. 2020), it might be necessary for new or adapted versions of programming environments to be designed to support learners’ existing culturally-based understandings.

Additionally, we noticed that several of the repeat gestures related to a special mapping of the code. A significant number of the participants explained the repeat by giving an example similar to that used in the learning session; some of the participants drew the arrow shown on the repeat block. Others represented it as a container similar to the repeat programming block with the other action blocks inside it. This was not surprising, as the
block-based programming language involved blocks with a visual representation written on them, affecting how the children understood and then simulated the repeat block.

There is likely to be an interaction between computing environments and computing metaphors, in which metaphors are used to design environments and then shape the metaphorical thinking of the users (Broaders et al. 2007). Our findings suggest that the design of educational materials in computing education could help learners by exploring conceptual metaphors and could be applied to inform further design guidelines, for example, the representation of iteration. This lens might help identify possible conceptual challenges, such as visually representing iteration and noise dynamics. These challenges might impact the user’s use of gestures when designing learning interfaces that involve gesture use to support learning. It is important to also note the culturally defined nature of some metaphors and consider the implications of children being asked to use computing environments whose metaphors clash with their existing conceptual metaphors.

In this study, gestures in which the hand moved along a linear axis typically referred to computing processes rather than direct constructs. This revealed that gestures mark both the start and endpoints of a process and a sequence of steps along a path. The delineated path axis was also interesting. The longitudinal axis corresponded with vertical lines of code. The traversal axis corresponded with the cultural left/right direction of time and the frontal axis appeared to correspond with a culture of time in relation to the body. Although gestures along these axes seemed to simulate processes, it was interesting to note the points along the trajectory because they often corresponded with algorithmic steps.

More work is needed to investigate the special metaphors that represent computing concepts and more investigation is needed to understand the potential benefits of explicitly encouraging gesturing, such as the gestures in this study, for making implicit knowledge explicit (GOV.UK 2013). In some countries, such as the UK, young children are expected to learn computing concepts from the age of five (Martha W. Alibali and Nathan 2007), so it would be worth investigating whether encouraging children to use particular gestures might support their understanding of computing. Additionally, in other domains, there is
evidence showing the benefits of teachers using specific gestures in the classroom (Valenzeno, Alibali, and Klatzky 2003) to improve students’ learning (Martha W. Alibali and Nathan 2012). There is emerging work in computing education, such as Solomon et al. who investigated the use of gestures by computer instructors (A. Solomon et al. 2020). Further investigation is needed to evaluate the anecdotal evidence.

Although this study was not designed to investigate the potential benefits of gestures, we noticed that gestures revealed information that participants did not express in words. Gesture use was beneficial for our understanding of the participants’ implicit knowledge. It can be used as a tool to facilitate understanding that was not expressed by words. For example, in our study, we spotted two instances where gestures revealed student knowledge and changed our classification of their understanding. This in line with Novack and Goldin-Meadow, who discovered that gesture reveals what learners know (Novack and Goldin-Meadow 2015).

5.4.3 RQ3.3 Are There Any Differences Evident in The Use Of Gestures and Conceptual Metaphors When Children Are Interviewed Again Two Weeks After The Introductory Programming Activity Vs. The First Interview?

The study identified one level 2 agreement with the metaphorical representation of noise and 13 solo representations (five from Interview 1 and eight from Interview 2). The repeat revealed three agreements, eight disagreements and six solo representations of the repeat concept across the interviews. Additionally, a single interview revealed five disagreements and one agreement with the repeat representation. Drawing a conclusion is tricky because of the small number of representation gestures that participants generated across the interviews. For instance, the participants might not have developed a full conceptual representation for the repetition. Thus, further work with more participants and thus a greater number of RG-metaphorical gestures could better assess any changes in the conceptual metaphors over time.

The results showed that participants generated more gestures in Interview 2. This may suggest that the participants needed time to more fully comprehend the concepts that they learned in Study 2. This finding could also support the notion of the use of gestures as a
tool to offload some cognitive load, as Wilson (2002) described in the use of the embodied cognition theory. Participants in Interview 1 were asked to explain the concepts immediately after they learned them: this process did not require much cognitive work as the concepts were new and, in the participants’, short-term memory. In Interview 2, the students needed to retain and offload the concepts from their long-term memories, which may have required more cognitive work. Another explanation might be the consolidation of the meaning of embodied representations in Interview 2. These gestural representations may have been easier to recall compared to symbolic linguistic representations after two weeks. Further work is needed to investigate possible explanation.

5.4.4 RQ3.4 What is The Effect of Interface Type (GUI Versus TUI) on The Use Of Spontaneous Gestures When Explaining Programming Concepts?

Study 3 took place following the completion of Study 2. In Study 2, participants programmed using either a TUI-PR or a GUI-SR. The results did not show a significant difference between the two interface groups in terms of the overall frequency gestures. These findings align with those of Study 1 and Study 2, which posit that object manipulation does not encourage gesturing.

5.5 IMPLICATIONS

The deliberate use metaphorical gestures in computing classes by teachers might improve the learning of abstract concepts such as program and iteration and encouraging learners to use gestures might help reveal misconceptions in learners’ knowledge that otherwise might not be picked up on. One of the most significant challenges CS teachers face with novices is helping them to build strategies and mental models to understand abstract concepts (Scopelitis, Mehus, and Stevens 2010), of which CS is full. There are other ways that gesture is used as a tool to convey knowledge. Scopelitis argues that gesture can be employed as a tool to build representations that the speaker and hearer can use to achieve a common understanding (Scopelitis, Mehus, and Stevens 2010). One participant (P37) said that the program was a ‘repeat’ while gesturing the form of a container, he described the meaning of a program with the help of a metaphorical gesture. Additionally, the participant expressed the concept of ‘put stuff inside’ by simulating putting one
programming block inside another programming block; he then expressed ‘then it will repeat’ by gesturing a circular motion. The participant was likely trying to embody and share their mental representation of a program, which gave the listener a concrete representation. This was very similar to Solomon’s finding in the case study of a CS instructor who used the gesture as a tool to convey the concept of a list (A. Solomon et al. 2020). In general, an educator’s awareness of their students’ thinking can be improved by paying attention to the gestures and by cautiously considering the reasons for these gestures and the conceptual mappings that they produce (Kendon 1997). Virtually all of the participants in the current study produced gestures that suggested hands-on or bodily simulation of the robot’s movement. Many of these gestures were not very precise about how the robot moved (for example, if it turned, it moved its head only, or turned and moved one step simultaneously). Such details of the robot’s movement being inaccurate might mean that students had limited understanding of how the code functioned. This could be addressed further.

5.6 Conclusion

This study explored the embodied expression of two programming concepts; program and iteration. The role and implication of embodied cognition have been studied extensively in STEM subjects, especially mathematics and science education. This study contributes to children’s computing education research by providing empirical evidence for the embodied nature of their understanding of computing concepts and draws attention to the potential benefits of exploring the role of embodied representations such as gestures in this domain. The findings show that the children aged six-to-seven years old produced less gestural evidence of metaphorical understanding compared to that of conceptual understanding. Additionally, this study provides supporting evidence for the ideas that gestures might have been used as a tool to offload cognitive load and as a secondary channel of communication instead of just using speech. This research also suggests how gestures might have been an indication of the embodiment of the children’s computing notions. The next chapter summarizes the main contributions of this thesis, its limitations and potential future research topics.
CHAPTER 6
CONCLUSION AND FUTURE DIRECTIONS

6.1 CONCLUSIONS

This thesis was motivated by the need for more empirical evidence about the efficacy of embodied learning approaches in computing education, given the international movement towards teaching computing at a younger age and acknowledging the potential associated with analysing embodied representations such as physicality, gestures and metaphors. Due to the novelty of the work in the field of computing, some of the empirical work has been exploratory. Specifically, this thesis aimed to develop our theoretical and practical understanding of how to teach programming concepts to younger children in an effective way. The findings presented in Studies 1, 2 and 3 form the main contributions of this thesis and addressed the following research questions.

RQ.1 ‘What are the key differences in how TUI and GUI programming environments support the development of programming skills for students aged six-to-seven years old?’

Many studies have advocated using tangibles in classrooms and teachers reportedly believe in their effectiveness, despite the fact that the available results are mixed and, furthermore, that they are not always easy to deploy for schools. RQ1 was answered in the first part of Study 1 and Study 2. Study 1 examined the role of physicality as an input to a programming environment, by comparing a TUI and GUI programming environment, with both the TUI and the GUI programming blocks controlling a PR as the output. In total, 42 students engaged in six learning activities using a TUI-PR or GUI-PR programming environment. In Study 2, the aim was to allow investigation of the importance of the physicality of the output of the system (the robot whose behaviour is
controlled by the programs created), by using an SR with the GUI environment and keeping the PR with the TUI environment. 50 students engaged in four learning activities using either a TUI-PR or GUI-SR programming environment. Studies 1 and 2 used a between-groups design and quantitative data were collected, including pre-test and post-test results, video recordings of the learning activities and ratings on attitudinal and enjoyment surveys.

**RQ1.1 ‘Are there any differences in learning outcomes, attitudes toward computing and enjoyment of computing?’** Improvements in learning were significantly higher for the GUI-PR group compared to the TUI-PR group, but post-activity gains in reported attitudes toward computing were significantly higher for the TUI-PR group, in Study 1. There was no difference in enjoyment scores, which were high for both groups. Study 2 reported no significant difference between TUI-PR or GUI-SR in terms of learning gains. However, the GUI-SR group experienced a significantly greater improvement in attitudes toward computing. Despite the difference in attitude change and the use of a graphical robot and tangible robot as different engaging outputs, no significant difference was found between enjoyment and the type of interface used for the activity. Furthermore, enjoyment of the interfaces for both groups showed similar levels compared to Study 1. These results have the potential to save schools from the challenges of deploying TUI-PRs, which can be costly compared to GUI-SRs (Sapounidis and Demetriadis 2017).

**RQ1.2 ‘Are there any differential gender effects in the above measures?’** No difference was found between genders in terms of learning gains, attitudinal change, or enjoyment in either Study (1 or 2).

**RQ.2 ‘How do young children use spontaneous gestures while learning programming and what is the relationship between gesturing and learning outcomes?’**

In Study 1 and Study 2, we examined the effect of different interfaces – TUI and GUI – on children's use of spontaneous gestures while completing programming tasks.
The answer to **RQ2.1** ‘**How does interface type affect the use of children's spontaneous gestures while completing programming tasks?**’ is that no significant difference was observed in the frequency of gestures between interface types in either study.

The answer to **RQ2.2** ‘**How are young children's learning outcomes related to the frequency of spontaneous gestures while completing programming tasks?**’ is that a statistically significant difference was identified between the mean learning gains in programming of high-frequency gesturers and low-frequency gesturers, with the top quartile showing significantly greater learning gains. Therefore, this research found an indication that gesture might either help learning or is a sign that the learning is actually happening. The exact relation and direction (if any) between gesturing and learning needs to be investigated in future work. Study 1 reported no correlation between the school-based mathematics scores that had been originally used to match the experimental pairs and the number of gestures they produced, suggesting that the high achievers in mathematics were not also frequent gesturers; in other words, that a high frequency of gesturing is not a standard feature of high achievers. Therefore, when a student who produces a high number of gestures also has a high learning gain in programming, this might be a small indication that the gesturing is helping the learning. This needs further investigation with a large data set and a more reliable second test such as a standardised test that indicates student ability. In Study 1, we did not find a correlation between the learning gains and the math scores, which might be because the scores provided by the teacher were not based on a standardised test: they were based on the teacher's judgment.

In contrast to Study 1, Study 2 found no statistically significant difference between the high- and low-frequency gesture groups in terms of learning outcomes. Nevertheless, all this left the answer to the question, ‘**How are young children's learning outcomes related to the frequency of spontaneous gestures while completing programming tasks?**’, rather uncertain.

As a result, an additional analysis of Study 1 data was conducted in Sections (3.4.2 and 4.4.3.3) to investigate the possible differences between the studies. First, in Study 2, no
significant difference was identified in learning gains between the two interface types. This may be attributable to the fact that the number of activities was reduced from six in Study 1 to four in Study 2, which occurred due to the time limitations associated with the school setting. The activities that were removed were the more complex activities that required children to embrace challenges and attempt new approaches to problem-solving. The absence of the kinds of challenge found in complex activities might have been the reason why no difference was identified in Study 2: namely, a floor effect, because the activities they solved were simple and did not reflect the differences between the interfaces.

Second, Study 1 found that attitudes to computing improved significantly more with the TUI-PR group, while in Study 2 it improved significantly more with the GUI-SR. This difference might be related to children’s perceptions of what counts as computing, i.e. more children might have considered that what they were doing was computing when they were using a GUI. Another possible explanation may be unanticipated additional effects of the different outputs. In Study 1, there was no statistically significant difference in session time or interaction time between the two conditions. However, in Study 2, the TUI-PR group spent less time interacting with the interface but took more session time. The session time, which included the time taken to understand the task, plan the solution and debug and execute the code, was significantly higher in the TUI-PR group compared to the GUI-SR group. The TUI-PR users needed longer sessions due to the additional time taken to set up the robot mat and for the physical robot to move while the GUI-SR had full control of the programming environment. The researcher did not need to intervene and adjust the activity setting or the robot as with the TUI-PR group. This could potentially have led to a greater sense of agency for the GUI-SR group compared to the TUI-PR group.

Third, the relationship of the frequency of gestures with learning gains was different in each study, which might be because of the difference in the number and complexity of activities. In Study 1, a difference was identified between learning gains in the high and low frequency gesture groups, which was not the case in Study 2 when two of the more
complex activities were excluded. Therefore, activity complexity may have played a role in children's different use of the two interfaces and also in stimulating their gestures when solving complex programming problems, which could significantly affect their learning.

RQ2.3. ‘What are the types and purposes of young children's spontaneous gestures while completing programming tasks?’ In both Study 1 and Study 2, participants used pointing gestures mostly to indicate objects or to simulate the code. The second most frequent type of gestures were iconic gestures (RG-literal), used to simulate the action of the robot (e.g., turning right or left or moving forward). Finally, the least frequently observed gestures were RG-metaphorical, which in all cases represented an abstract concept (iteration).

RQ2.4 ‘How are gestures used to represent abstract programming concepts (iteration)?’ In Study 1 and Study 2 a qualitative analysis of representational gestures revealed that certain children used spontaneous hand gestures to demonstrate abstract computational concepts such as iteration. This demonstrated that young children can generate representational gestures spontaneously when discussing and completing a programming task. Additionally, this research shows that the children displayed similar gestures to adult learners, as reported in (Manches et al. 2020) when CS students were asked to represent the concept of iteration.

RQ.3 The question ‘How are conceptual metaphors used by young children to explain programming concepts?’ was answered in Study 3.

In Studies 1 and 2, it was found that only a small number of children used conceptual metaphors when solving programming tasks and so it was of interest to investigate further children's use of conceptual metaphors to explain computing concepts. In particular, this research question sought to address a gap in current theoretical understandings of computing education by drawing on embodied cognition theory. Drawing on methodological tools from cognitive linguistics and gesture research, we analysed how primary school students used spontaneous co-speech gestures when responding to
interview questions and describing programming concepts. With a sample group of 45 primary school students in Saudi Arabia aged six-to-seven years, we conducted two interviews at two different times (once directly after a programming activity and once approximately two weeks later). We analysed their responses when asked to explain two programming concepts: program and iteration.

RQ3.1 ‘What types of spontaneous gestures do young children use to explain programming concepts after an introductory programming activity?’ The participants generated RG-literal gestures most commonly, followed by pointing gestures and then RG-metaphorical gestures. In Study 3, the relevant stimuli (i.e., written code) were absent in most of the questions (Q1, Q2 and Q4). The only relevant stimulus that was available was the code that participants were asked to explain in Q3 and Q5. This likely explains the limited use of pointing gestures among the participants compared with Studies 1 and 2. The participants used RG-literal gestures most commonly when the code was in front of them. RG-metaphorical gestures were the least used gesture type, which might be because metaphorical thinking (i.e., abstract thinking) does not spontaneously play a strong role in reasoning about CS concepts at the age of six-to-seven years. This is not surprising because the use of metaphors of this kind is sophisticated for young children of this age.

RQ3.2 ‘What types of conceptual metaphor do young children use to explain programming concepts?’ after an introductory programming activity. Participants who used metaphorical gestures drew upon two overarching embodied metaphors in their explanations. In the first of these, for computing constructs as physical objects, participants simulated manipulating an object (pinching) when referring to programming code or the hand acting as a boundary of a physical object representing a container. In the second case, for computing processes as a motion along the path, participants moved their hands along one of three body-based axes (longitudinal, transverse and frontal) when referring to chronological sequences. The findings show a pattern from these gestures, thereby suggesting the potential of this methodological approach to provide teachers with
a deeper understanding of the nature of learner cognition in the domain of computing education.

RQ3.3 ‘Are there any differences evident in the use of gestures and conceptual metaphors when children are interviewed again, two weeks after the introductory programming activity and first interview?’ Drawing a conclusive answer is difficult because of the small number of RG-metaphorical gestures that were generated across both the interviews (only 12 RG-metaphorical gestures in total). However, the quantitative data showed that participants generated significantly more gestures in Interview 2 (where the average number of gestures was 12.98) than in Interview 1 (where the average number was 10.30). The reason for this is unclear. It could suggest that participants relied more on gestures as a support in the second interview when the concepts were less fresh in their minds. This could be indicative of the use of gestures as a tool to offload cognitive load. Participants in Interview 1 were asked to explain the concepts immediately after they had learned them. This process did not require much cognitive effort as the concepts were fresh in the participants' short-term memory, while in Interview 2, the students needed to find the information from their long-term memories, which may have required more cognitive effort, potentially suggesting that gesture was used to offload this cognitive work.

RQ3.4 ‘What is the effect of interface type (GUI versus TUI) on the use of spontaneous gestures when explaining programming concepts?’ There was no significant difference between the two interface groups in terms of overall frequency of gestures. These results are in line with Study 1 and Study 2, which posits that object manipulation does not encourage gesturing.

6.2 Contributions

This dissertation’s main contributions relate to three important embodiment approaches: the physicality of the interface, the use of gestures and the use of conceptual metaphors. The contributions fall under two different categories: interaction design-related
contributions pertaining to the importance of physicality in programming environments for children and contributions to the understanding of the importance of gesture and conceptual metaphors in primary school CS education.

6.2.1 Interaction Design-related Contributions

This doctoral research adds to the developing literature in interaction design for CS education concerning the importance of the physicality of interfaces for children aged six–to-seven years, two of which are mentioned below:

Contribution 1: Evidence that GUIs can have benefits over TUIs in programming environments for young children.

Despite the common belief that TUIs are important for teaching computing to young children, this doctoral research illustrates how GUIs can be as effective as TUIs and, in some cases, even more effective in increasing learning outcomes. GUIs were associated with positive attitude changes toward computing and a high rate of enjoyment. The setup time for TUIs and in particular physical output devices such as robots, often requires the involvement of an expert adult in between activities, which can be distracting and disempowering for children. This research showed that with a GUI and an on-screen robot, children were able to dedicate more time to the task itself and engage in active learning. This finding provides an empirical basis for encouraging schools to avoid the challenges of deploying TUI-PRs (i.e., physical robots), which cost more than GUI-SRs (i.e., on-screen robots).

Contribution 2: Investigation of the possible relationship between increased embodiment in the interface and the output device and increased use of embodied representations of concepts through gestures, with no relationship found across two studies.

Given the mixed findings in previous work relating to the benefits of TUIs, this work investigated one plausible mechanism by which the physicality of interface and output devices might influence learning: the prompting or impairing of gesturing. Despite TUIs
requiring embodied interaction, this research showed no evidence of an increase or decrease in the use of embodied expression of concepts (i.e., gestures) when learners used a TUI compared with when they used a GUI. There was also no evidence of the physicality of the output device leading to an increase or decrease in the use of gestures.

6.2.2 Gesture and Conceptual Metaphor-related Contributions

This doctoral research adds to the literature on the use of spontaneous gestures in CS education by extending the investigation to young children aged six-to-seven years for the following points:

**Contribution 3: Description and identification of children’s use of spontaneous gestures when solving programming tasks and explaining programming concepts.**

In line with previous work that explored the use of gestures in CS by older learners, the studies in this thesis found that children use concrete gestures (e.g., pointing, literal) and metaphorical gestures. This research confirms that the children used similar types of gestures and conceptual metaphors compared to adult learners. However, children mostly produced concrete gestures and few metaphorical gestures.

**Contribution 4: Demonstrated how studying children’s gestures can help characterise learner conceptions in primary computing, potentially allowing identification of misconceptions and aiding the identification of productive pedagogic strategies.**

Addressing the gap in the theoretical evidence base for understanding of learning in CS, this research used an embodied cognition lens to promote understanding of the development of CS and potentially approaches that draw upon embodied cognition to inform CS. The findings showed a pattern in children’s gestures (e.g., circular gestures represented repeat), thereby suggesting the potential of using embodied cognition lens as a methodological approach to provide educators with a deeper understanding of the nature of learner cognition in the domain of CS education. For example, in the *Tica* programming language used in this thesis, repeat meant that enclosed actions will be executed twice.
When children gestured circular movements more than two times this might have suggested they did not fully understand the functionality.

**Contribution 5: Evidence that children use spontaneous hand gestures to demonstrate abstract computational concepts, even in the absence of relevant stimuli (i.e., written code).**

Similar to previous work that explored the use of gestures in computing education with older learners, this research found that children used abstract computational concepts in the absence of relevant stimuli. This indicates that gestures may reflect the embodiment of the children’s knowledge of computing concepts and points to the embodiment of children's offline thinking in the computing domain.

**Contribution 6: Evidence that the direction of written language in a culture affects the direction/use of conceptual metaphor in CS**

This research is the first to investigate the use of conceptual metaphors in learners whose first language is a right-to-left language (i.e., Arabic) and the first to present tentative evidence of cultural influences on embodied conceptualisations, with right-to-left gestural loops common despite the programming language used implying left-to-right order. The findings indicate that when explaining programming concepts, the gestures used by these learners are notably different in terms of the aspect of their direction compared to the gestures used by learners whose first language is a left-to-right language (e.g., English). The features of languages impact human conceptualisations of their surroundings, language may influence thought in a clear semantic way. Therefore, to increase the accessibility of CS education for a non-left-to-right language speaker, new or adapted versions of programming environments might be needed to support learners' existing culturally based understandings.

The programming blocks in the ‘Tica’ language used in this research are placed from top to bottom, but there are some elements of the design that imply left-to-right reading order. The blocks have text written in English, are slightly aligned with the code’s left to right,
and the arrow on the repeat blocks moves in the clockwise direction. This might create a potential conceptual challenge for children if right-to-left reading is their main model. Although the children who participated in this research were studying at an international school that teaches English as the main language, and the programming language implied a left-to-right direction, the cultural context in which the ideas are presented appeared to be strongly present.

**Contribution 7: Evidence of a positive relationship between the mean learning gains of high-frequency gesturers and low-frequency gesturers when the tasks had varying problem difficulties.**

In line with similar studies in mathematics related to the link between spontaneous gestures and learning, this research demonstrated an association between use of gestures and learning outcomes in computing. Study 1 found that the top quartile of high-frequency gesturers when attempting to solve programming problems showed significantly greater learning gains than the bottom quartile, indicating that gesture might either help learning or is a sign that the learning is happening. Future work is needed to investigate the relationship in more depth and investigate a potential causal link.

**6.3 LIMITATIONS**

Several limitations have been identified within the studies that make up this thesis, as discussed below.

1. The time was limited when students could be taken out of their classroom and everyday work. In Study 2, the learning session time was reduced from 45 minutes to 30 minutes by removing two more complex learning activities. This might have affected the results and made it difficult to draw conclusions (see Section 4.4.3.2).

2. Although metaphors are used extensively in CS, such as when programming data structures (e.g., objects, variables, stacks, queues, linked lists, trees and pipes), the studies were limited to the relatively simple concepts of program and iteration due to our target users (i.e., primary school children).
3. In this dissertation, gesture data was analysed manually using psychological analysis methods. However, using motion detection systems to collect and analyse videos might allow future researchers to benefit from significantly larger and richer datasets.

4. The children's learning measurements were limited to their immediate effect rather than by considering the mid-to-long-term impact or the effect on children's abilities to use the concepts they learned in an out-of-class project.

5. This research was limited to the evaluation of specific learning outcomes as set out in the UK's Key Stage 1 curriculum, but in the context of an international school in Saudi-Arabia.

6. This research was limited to investigating learners' use of gestures; however, the literature shows promising results for the meaningful use of gestures by teachers based on how it supports learning in other fields. More about this implication and suggestions for further work regarding investigation of teacher gestures can be found in section 6.4.4.

7. Even though the children were enrolled in the same first grade class, they differed in their ability to communicate effectively and engage in turn-taking behaviours that advance the common goal based on their development. Thus, failure to learn and failure to use gestures may both reflect a particular child's younger development rather than supporting evidence of gestures leading to greater comprehension. Children’s ability to communicate by speech or gestures might be considered in the pre-test in future work, when further investigating of children gestures will be addressed.

8. The average sample size in the three studies was 47 participants, which was appropriate for this exploratory context However, it did not allow for conclusive statements about the effectiveness of the TUI versus GUI or spontaneous gestures, which would require a larger sample size.
6.4 Implications and Future Work

The work presented in this thesis fits into a broader context of the new challenges and opportunities that have emerged due to the novelty in the field of computing and the significant interest and investment in computing education worldwide. The findings of this thesis have a number of implications and ideas for future research, including those provoked by the following:

6.4.1 Interfaces for Learning Programming

- **Choice of interface type in the classroom:** The literature indicates that teachers use tangibles to increase the learnability of abstract concepts. However, having a tangible programming environment in the classroom represents a challenge in terms of its deployment and maintainability. This research compared a GUI and a TUI as input and an engaging robot as output, given that the type of interface used when learning programming can influence the development of computing skills (Rijo-Garcia, Segredo, and Leon 2022). The data showed that both kinds of interface could be of some benefit, so a choice between them depends on the primary goal of educators, e.g. learning gains vs. enjoyment. The group that displayed the greatest learning gains was the GUI-PR. This group used a familiar interface and engaging output, which is potentially the optimal combination in terms of learning; this is because the touch-screen environment is easy for children to use, and they physically observe the physical robot moving to the given goal. In addition, these systems are commercially available and easy to adapt in the school context, examples of which include LEGO® MINDSTORMS®. On the other hand, the TUI-PR group showed a greater improvement in attitude toward computing than the GUI-PR group, while there was no significant difference in interaction time (i.e., the time that children spent on the interfaces). This suggests that if teachers can find a mature TUI system in the market, they could use it to improve children’s attitudes toward computing. Noteworthily, however, GUI-SR also showed a greater improvement in terms of children’s attitudes toward computing than the GUI-PR group, and these systems are widely available in
schools. Taken together, the results indicate that all of the interfaces showed a high enjoyment rate.

**Depth and transfer of learning:** This dissertation measured the effects of interface on children's learning in terms of its immediate effect rather than by considering the mid-to-long-term impact or the effect on children's ability to use the concepts they had learned in an out-of-class project. Therefore, further research is needed to examine the possibilities of long-term retention, far transfer and the question of how children can apply knowledge to solve real programming problems rather than paper tests. Also, a 30 or 40-minute learning session is a short intervention, and a longitudinal study would be required to determine whether the effects of interface type are the same across a longer intervention.

**Research in other national contexts and improving learning through enjoyment and attitudes:** In our work, we evaluated specific learning outcomes as set out in the UK's Key Stage 1 curriculum, but in the context of an international school in Saudi-Arabia. Further research is needed to determine the extent to which each type of user interface is suited to learning specific concepts in the curricula of other countries. Additionally, computer programming for young children is still a recent development and not all countries teach the subject at the primary level. Therefore, further research should address the potential cultural differences in learning patterns, debugging styles and gesture types. This would help educators to provide beneficial educational materials and learning experiences for young children who are exploring programming concepts. Additionally, it may be possible to enhance the quality of learning by studying enjoyment and attitudes toward computing. This might affect children when they come to decide whether to pursue a career in computing.

**6.4.2 The Role of Gestures in Learning, and Problem-Solving**

**Activity complexity and the use of gestures:** The results in Section 3.4.3.2 suggest that a higher frequency of spontaneous gestures may be associated with solving more complex cognitive tasks, as was found in Study 1 when children solved the complex task ‘Task 3’. This research cannot and did not set out to
explicate exactly what role gesturing might play in problem-solving and learning, but there are some indications that gesturing plays a positive role in both processes. This is worth investigating further to identify (i) whether allowing vs. inhibiting spontaneous gesturing has an effect on the quality of learning and problem-solving, (ii) whether there are individual differences in terms of the amount of spontaneous gesturing, (iii) whether spontaneous gesturing is a way of off-loading some of the complexity of cognitive tasks, and (iv) if there are any features of activities that encourage children to use their gestures effectively to assist their learning and problem-solving. Research in this area could potentially aid educators in developing an understanding of how best to integrate and encourage spontaneous gestures in the classroom for tasks involving complex activities.

**Encouraging actions (gesturing and body):** The data collected in this research consisted of spontaneous gestures. A positive relationship was identified between the number of gestures generated and the amount of learning. This reflects the possibility that when teachers encourage children to use gestures, this can lead to both short-term and long-term benefits (Cook, Mitchell, and Goldin-Meadow 2008; Cook, Yip, and Goldin-Meadow 2010). Further research is needed to examine the effect of prompting gestures on children’s understanding. Although this research was limited to investigating the role of gestures, whole-body movement was also observed, which can be used to represent ideas in a similar way as gestures. This research presented several examples of children using their whole body to represent iteration or to represent noise through dancing. Gestures can be linked to the work related to body-based action and embodied learning, such as CS Unplugged (Bell, Lambert, and Marghitu 2012), based on the congruence between these embodied activities and computing concepts (Skulmowski and Rey 2018). This research is an encouraging sign for future researchers, showing the value of reflecting on the mapping between specific body actions and computing concepts, as well as the potential to adopt these actions through gestures, especially given the way that gestures can connect action, experience, and scientific language (Roth 2001).
- **Depth and transfer of learning**: research in mathematics education (Novack et al. 2014) indicates that the use of representational gestures can lead to differences in the depth and transfer of learning. Although Study 3 examined gestures through two different interviews, more work is needed to examine how gesturing evolves with learners' ability over longer time. For example, does gesturing decrease over time as concepts get more established or does it increase over time as it seems to aid understanding? Future work is needed to examine how certain gestures are related to students' learning and how gestures can support different learning needs. This could be important in evaluating the potential of teaching gestures to assess and support learning (Manches et al. 2020).

6.4.3 **The Role of Gestures in Communication and Explaining**

In this research, we observed that students generated spontaneous gestures in Study 1 and Study 2 when they were solving programming tasks in pairs and also in Study 3 when they were explaining programming concepts to an adult (in this case, the doctoral researcher). A question for the future is the extent to which spontaneous gesturing accompanying a child’s explanation is more about (i) creating an explanation from an embodied concept, or (ii) more about helping the child to generate a coherent explanation, or (iii) more about the child trying to make her meaning clearer to the other, or (iv) a mixture of all of these?

6.4.4 **The Role of Metaphors in Communication, Learning and Explaining**

- **The role of metaphoric gestures as a tool to facilitate communication for both the gesture producer and the gesture receiver**: In this research, we observed that students used gestures in Study 1, Study 2 and Study 3. However, this research did not focus on the issue of exactly when children used metaphorical gestures or their reasons for using them. A similar set of questions arise as in the previous subsection. For example, do children feel that they spontaneously need to use metaphors in the form of gestures rather than using just speech to represent their understanding of a complex idea such as iteration? Or do children decide to use metaphorical gestures because they are talking to an adult whom these gestures may assist? Future research should explore the
intention associated with using metaphorical gestures, enabling educators to use these gestures to identify any misconceptions that students might have and correct their understanding.

- **The relationship between the development of metaphor and understanding:** Even though in Study 3 we observed the gestures before and after a two-week gap, we did not trace a developmental path in the children's gestures and their relationship with understanding. Future work can look at how gestural metaphors change as children develop their computing knowledge.

- **Expand the taxonomies of gesture and use these in learning sessions:** Further work could examine the role of metaphors in computer pedagogy more broadly, thereby helping to inspire the creation of new tangible devices or, in any case, the examination of topics of interest in didactics (e.g., which gestures to make when teaching and what metaphors are related to these gestures).

### 6.4.5 Teachers and Gestures

Teachers can use gestures in computing education to aid students in understanding, reasoning about, and communicating computing concepts.

- Teachers’ spontaneous use of gestures: Solomon and colleagues (2020) showed that teachers used gestures naturalistically when they taught computing in the classroom (A. Solomon et al. 2020) which is in line with in other domains (Alibali and Mitchell 2007). Although this doctoral research did not investigate whether the teachers gesture in computing classroom were naturalistic, future work is needed to investigate teacher spontaneous gestures in order to help computing education community to understand how embodiment supports computing learners.

- Teachers’ use of guided gestures: In mathematics, there is evidence indicating that teachers regulate their gestures to scaffold learner understanding (Valenzeno, Alibali, and Klatzky 2003). Therefore, further work is needed to explore computing teachers’ use of gestures by engaging them to reflect on their gestures and training them to use guided gestures, similar to their training to use particular visual representations or technology.
- **Teachers’ observations of gestures**: The research highlights that many children use hand gestures to explain and communicate programming concepts. This finding underlines the relevance of gestures in communicating an understanding of computing topics among both teachers and students. The children used hand gestures to explain and communicate programming concepts in all studies. When teachers observe students’ gestures in the classroom, this can enable them to assess the understanding of learners (Kelly et al. 2002). This work documents and describes students’ use of gestures while solving a programming task or explaining computing concepts. Future research might seek to evaluate how teachers can increase the attention they pay towards children’s use of gestures in the classroom, and how to assess the use of gestures. Since gestures can serve as a way to visualize a learner’s understanding, teachers may be able to identify and assess potential strengths and weaknesses by considering them.
REFERENCES


Almjally, Abrar, Kate Howland, and Judith Good. 2020a. “Comparing Tuis and Guis
for Primary School Programming.” In *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE*, Association for Computing Machinery, 521–27.


Conference on Creativity and Cognition, 275–84.


Asharq Al-awsat. 2006. “اعتماد منهج الحاسب الآلي كمادة أساسية للصف الأول الثانوي للبنات.”


Bell, Tim, Lynn Lambert, and Daniela Marghitu. 2012. “CS Unplugged, Outreach and CS Kinesthetic Activities.” In New York, NY, USA: Association for Computing Machinery (ACM), 676.


Joint Council for Qualifications. 2012. *Provisional GCE A Level Results - 2012 (All UK Candidates)*.


Kayan-Fadlelmula, Fatma, Abdellatif Sellami, Nada Abdelkader, and Salman Umer. 2022. 9 International Journal of STEM Education *A Systematic Review of STEM*


Scherer, Ronny, Fazilat Siddiq, and Bárbara Sánchez Viveros. 2020. “A Meta-


Strawhacker, Amanda, and Marina U Bers. 2015. “‘I Want My Robot to Look for Food’: Comparing Kindergartner’s Programming Comprehension Using


APPENDIX A

Test A

1. Look at the picture below (the arrow shows where the robot is, and which way it is facing):

A. What shape will the robot do to get to the Finish square?

B. How many moves to the Right will the robot have to do to go to the Finish square?

0 1 2 3

C. What are the first 2 moves the robot will have to do to go to the Finish square?

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2. Which of the programs below will get the robot to make noise and go to the Finish square?

3. Which square of the grid will the robot go to, when following the program below? Use the given sticker to show which will be the Finish square.
4. Write a program to make the robot go to the Finish square. (Use the given stickers).

<table>
<thead>
<tr>
<th>4</th>
<th>Solution 1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>You can use as many stickers as you need</td>
</tr>
</tbody>
</table>

![Diagram of a grid with a start and finish marker]
5. Now make the robot go to the Finish square using 5 blocks only. (Use the given stickers).

6. Which solution is more advanced/better and why?

Solution 1

Solution 2

Why ..................................
Test B

Participant number: ________________________________

1. Look at the picture below (the arrow shows where the robot is, and which way it is facing):

![Robot Diagram]

A. What shape will the robot do to get to the Finish square?

![Shape Options]

B. How many moves to the Left will the robot have to do to go to the Finish square?

0 1 2 3

C. What are the first 2 moves the robot will have to do to go to the Finish square?

![Robot Move Options]
2. Which of the programs below will get the robot to make noise and go to the Finish square?

3. Which square of the grid will the robot go to, when following the program below? Use the given sticker to show which will be the Finish square.

Program:  

Solution:
4. Write a program to make the robot go to the Finish square. (Use the given stickers).

<table>
<thead>
<tr>
<th>4</th>
</tr>
</thead>
</table>

**Solution 1**

You can use as many stickers as you need
5. Now make the robot go to the finish square using 5 blocks only. (Use the given stickers).

![Diagram of a grid with 'X's and 'S' indicating the start and finish squares.]

6. Which solution is more advanced/better and why?

Solution 1

Solution 2

Why ..................
APPENDIX B

Attitudinal survey

Participant number: ________________________________

Pleas circle the smiley face that match you’re feeling about the statement

I like computing

I am good at computing

Like the challenge of computing

Computing is fun

I want to find out more about computing

Enjoyment survey

Participant number: ___________________________________________

Please circle the smiley face that match you’re feeling about the question

How much did you like the tool (circle one face?)

I did not like it at all  I did not like it  It was ok  I like it  I liked it very much

Would like to try it again?

No  Maybe  Yes

Would you tell your friend about the tool?

No  Maybe  Yes

APPENDIX C

Ministry of Education ethical approval
Study 1 Head Teacher’s Consent Sheet

HEAD TEACHER’S CONSENT SHEET

Researchers are required to abide by ethical guidelines when working in schools. These cover topics such as gaining appropriate consent, permitting children to withdraw from the study and keeping data confidential. We would be grateful if you could check and sign the following sheet to show that you approve of the research procedures for this study.

PROJECT NAME
Investigating the use of TUIs and GUIs for 6 to 7 year olds learning programming

APPROVED BY
University of Sussex Sciences & Technology Cross-Schools Research Ethics Committee (crescitec@sussex.ac.uk)

NAME OF RESEARCHER VISITING THE SCHOOL
Abrar Almjally (a.Almjally@sussex.ac.uk)

NAME OF PROJECT SUPERVISORS
Dr. Kate Howland (k.l.howland@sussex.ac.uk), Prof Judith Good (J.Good@sussex.ac.uk)

PERIOD OF VISITS
Autumn Term 2018

CLASSES/YEAR GROUPS VISITED
Year 1

PROJECT DESCRIPTION
We wish to invite your students to take part in a research study to further our understanding of the advantages and disadvantages of common approaches to teaching programming to primary-school students and explore the relationships between interface types, attitudes towards computing, engagement and gender. We are focusing on comparing two popular interfaces, a graphical user interface (GUI) that uses touch-screen programming blocks and a tangible user interface (TUI) that uses physical programming blocks. Currently there is an interest in tangible approaches to supporting young children as they learn programming, but there has been mixed evidence as to their learning benefits. We will be using activities which are adopted from the United Kingdom’s Computing at School ‘Quick Start’ Key Stage 1 repository of programming learning activities. In the study, some students may manipulate 3D-printed computer blocks in front of tablets (the TUI) and the other students will manipulate virtual blocks on a tablet application (GUI). All students will use either tangible or virtual blocks to manipulate a physical robot suitable for ages 6+ [mBot robot] and engage in paper-based activities.

PROCEDURE
The research will take place in a quiet room within your school. We aim to test 40 children, with 20 children in each group. First, children will complete a pre-test and attitudinal survey before they begin the practical session. Second, the researcher will randomly assign the children to pairs and give them one-to-two instruction on the functionality of either the TUI or GUI. The students will complete six short activities and answer a post-test followed by an attitudinal survey to assess their attitudes towards computing and enjoyment surveys to assess their feeling about the tool. We will video record the session by using a
camera positioned in such a way as to capture children’s spontaneous gestures and their facial expressions to gather insight about their enjoyment. Each session should last approximately 30 to 40 minutes.

CONSENT
Parents/carers will be sent an information letter with details of the study and a permission form to be signed and returned in order for their child(ren) to participate in the study. Parents/carers should be given at least one week to read and respond to this letter. Children will be asked for verbal consent to participate in the study after receiving an initial briefing on the nature of the study and the procedures involved. Children will also be given the option to withdraw from the study at any point.

Please sign below to confirm that you:

- Understand the requirements of children who take part in the research
- Have received detailed descriptions of the methods and materials to be used
- Give approval for the research to take place at your school

Name of school:  

Name of [Head Teacher]:  

Signature:
Study 1 Information Consent Sheet

INFORMATION & CONSENT SHEET

PROJECT NAME
Investigating the use of TUIs and GUIs for 6 to 7 year olds learning programming

INVITATION TO TAKE PART

Your child is invited to take part in a research study to better understand the advantages and disadvantages of common approaches to teaching programming to primary-school students and explore the relationships between interface types, attitudes towards computing, engagement and gender. We are currently focusing on comparing two popular interfaces, a graphical user interface (GUI) that uses touch-screen programming blocks and a tangible user interface (TUI) that uses physical programming blocks. Currently there is an interest in tangible approaches to supporting young children as they learn programming, but there has been mixed evidence about their learning benefits.

Thank you for carefully reading this information sheet, a copy of which you can keep for your records. This study is being conducted by student researcher Abrar Almjally from the School of Engineering and Informatics, University of Sussex, UK who is happy to be contacted if you have any questions (a.almjally@sussex.ac.uk). The University of Sussex has insurance in place to cover its legal liabilities in respect of this study

WHY HAS MY CHILD BEEN INVITED FOR TESTING AND WHAT WILL THEY DO?

We are expecting to test 40 children from year 1 of School. The session should last approximately 30 to 40 minutes and be take place at School during normal hours. We will attempt to measure the interface effect on learning outcomes and attitudes, with each child being asked to complete a short pre-test and an attitudinal survey before session phase. In the session, your child will be assigned to work in pairs and given a one-to-two instruction on one of the interfaces (either the TUI or the GUI) to program a robot, complete six short programming activities and answer a post-test, attitudinal
survey to assess their attitudes towards computing and enjoyment surveys to assess their feelings towards the tool. The session will be video recorded using a camera positioned in such a way as to capture children’s spontaneous gestures and their facial expression to gather insight about their enjoyment.

ARE THERE ANY RISKS TO TAKING PART?
During the session, your child will interact with physical and paper-based learning materials, all of which have been approved by the School. The learning activities will pose no additional risk to your child’s safety.

WHAT WILL HAPPEN TO THE RESULTS AND MY CHILD’S PERSONAL INFORMATION?
The results of this research may be written into a scientific report for an informatics PhD thesis and/or publication. We anticipate being able to provide a summary of our findings on request from 01/04/2019 (a.almjally@sussex.ac.uk). Your child’s anonymity will be ensured in the way described in the consent information below. Please read this information carefully and then, if you wish for your child to take part, please complete the personal information questions and sign to show you have fully understood this sheet and that you consent for your child to take part in the study as it is described here.

CONSENT

• I understand that I am agreeing to allow my child to take part in the University of Sussex research described here and that I have explained the project to my child and I have read and understood this information sheet.

• I understand that my child’s participation is entirely voluntary, that they can choose not to participate in part or all of the study and that they can withdraw at any stage of testing without having to give a reason and without being penalised in any way.

• I understand I can request without penalty that my child’s data be withdrawn and deleted even after testing is complete, any time up until the results are analysed (01/1/2019).

• I consent to the processing of my child’s personal information for the purposes of this research study. I understand that such information will be treated as strictly confidential (subject to legal
limitations) and handled in accordance with the General Data Protection Regulation (GDPR) 2016.

- I understand that my child’s collected data will be stored in a de-identified way (e.g., using ID numbers not names) and kept separate from other details about my child (e.g. from the consent form). Electronic data will be stored in a password-protected file and hard-copies will be stored in a locked drawer. De-identified data may be made publicly available through online data repositories or at the request of other researchers.

- I understand that my child’s identity will remain confidential in any written reports of this research and that no information I disclose will lead to the identification in those reports of any individual either by the researchers or by any other party, without first obtaining my written permission.

- I understand that my child’s name and data will not be shared with any third party outside the research group, unless I later provide written permission.

- I consent that the session will be audio/video recorded and understand that the recordings will be stored anonymously, using password-protected software and will be used for specific research purposes.

- Please initial this box if you consent to the use of video material and images from your child’s participation in this study at academic conferences and research publications. (Note, Your child can participate in the study even if you do not initial this box)

- Please initial this box if you consent to share the video material and images from your child’s participation in this study with other
interested researchers. (Note, Your child can participate in the study even if you do not initial this box)

Name of Child: ______________________________________

Name of Parent or Guardian: __________________________

Date: ______________________________________________

Age of Child: _______________________________________

Signature: _________________________________________
Certificate of Approval

<table>
<thead>
<tr>
<th>Reference Number</th>
<th>ERAA21381</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title Of Project</td>
<td>Investigating the use of TUIs and CUIs for 0 to 7 year olds programming learning</td>
</tr>
<tr>
<td>Principal Investigator (PI):</td>
<td>Kate Howland</td>
</tr>
<tr>
<td>Student</td>
<td>Abrar Ninjally</td>
</tr>
<tr>
<td>Collaborators</td>
<td></td>
</tr>
<tr>
<td>Duration Of Approval</td>
<td>2 months</td>
</tr>
<tr>
<td>Expected Start Date</td>
<td>10-Sep-2018</td>
</tr>
<tr>
<td>Date Of Approval</td>
<td>03-Aug-2018</td>
</tr>
<tr>
<td>Approval Expiry Date</td>
<td>10-Nov-2018</td>
</tr>
<tr>
<td>Approved By</td>
<td>Karen Long</td>
</tr>
<tr>
<td>Name of Authorised Signatory</td>
<td>Karen Long</td>
</tr>
<tr>
<td>Date</td>
<td>03-Aug-2018</td>
</tr>
</tbody>
</table>

*NB. If the actual project start date is delayed beyond 12 months of the expected start date, this Certificate of Approval will lapse and the project will need to be reviewed again to take account of changed circumstances such as legislation, sponsor requirements and University procedures.

Please note and follow the requirements for approved submissions:

Amendments to protocol
* Any changes or amendments to approved protocols must be submitted to the C-REC for authorisation prior to implementation.

Feedback regarding the status and conduct of approved projects
* Any incidents with ethical implications that occur during the implementation of the project must be reported immediately to the Chair of the C-REC.

Feedback regarding any adverse(1) and unexpected events(2)
* Any adverse (undesirable and unintended) and unexpected events that occur during the implementation of the project must be reported to the Chair of the Science and Technology C-REC. In the event of a serious adverse event, research must be stopped immediately and the Chair alerted within 24 hours of the occurrence.

Monitoring of Approved studies
The University may undertake periodic monitoring of approved studies. Researchers will be requested to report on the outcomes of research activity in relation to approvals that were granted (full applications and amendments).

Research Standards
Failure to conduct University research in alignment with the Code of Practice for Research may be investigated under the Procedure for the investigation of Allegations of Misconduct in Research or other appropriate internal mechanisms (3). Any queries can be addressed to the Research Governance Office: rgooffice@sussex.ac.uk

(1) An “adverse event” is one that occurs during the course of a research protocol that either causes physical or psychological harm, or increases the risk of physical or psychological harm, or results in a loss of privacy and/or confidentiality to research participant or others.

(2) An “unexpected event” is an occurrence or situation during the course of a research project that was a) harmful to a participant taking part in the research, or b) increased the probability of harm to participants taking part in the research.

(3) http://www.sussex.ac.uk/staff/research/ri.policy/research-policy
HEAD TEACHER'S CONSENT SHEET

Researchers are required to abide by ethical guidelines when working in schools. These cover topics such as gaining appropriate consent, permitting children to withdraw from the study and keeping data confidential. We would be grateful if you could check and sign the following sheet to show that you approve of the research procedures for this study.

PROJECT NAME
Interfaces and spontaneous gestures in primary programming education

APPROVED BY
University of Sussex Sciences & Technology Cross-Schools Research Ethics Committee (crecscitec@sussex.ac.uk)

NAME OF RESEARCHER VISITING THE SCHOOL
Abrar Almjally (a.Almjally@sussex.ac.uk)

NAME OF PROJECT SUPERVISORS
Dr. Kate Howland (k.l.howland@sussex.ac.uk), Prof Judith Good (J.Good@sussex.ac.uk)

PERIOD OF VISITS
Autumn Term 2019

CLASSES/YEAR GROUPS VISITED
Year 1

PROJECT DESCRIPTION
We wish to invite your students to take part in a research study to further our understanding of the advantages and disadvantages of common approaches to teach programming to primary-school students. The study consists of two parts. Part 1 aims to investigate the role of physical manipulation in learning programming. To do this we will compare a tangible user interface (TUI) that uses physical programming blocks and controls a physical robot, with a graphical user interface (GUI) that uses touch-screen programming blocks and controls an on-screen robot. We will be using activities, which are adopted from the United Kingdom's Computing at School 'Quick Start' Key Stage 1 repository of programming learning activities. Our objective is exploring the relationships between interface types, attitudes towards computing, enjoyment and gender. In this part, some students may manipulate 3D-printed computer blocks in front of tablets (the TUI) manipulate a physical robot suitable for ages six+ [mBot robot] and the other students will manipulate virtual blocks and virtual robot on a tablet application (GUI). All the students will engage in paper-based activities and complete surveys, pre and post-tests. Part 2 aims to investigate children's
spontaneous gestures when explaining programming concepts. There is evidence for the importance of gestures in other fields of learning, but it is still emerging in the computing education field.

**PROCEDURE**
The research will take place in a quiet room within your school. We aim to run the study with 50 children from year 1 of school, with 25 children in each group. Part 1, first children will complete a pre-test and attitudinal survey before they begin the practical session. Second, the researcher will randomly assign the children to pairs and give them one-to-two instruction on the functionality of either the TUI or GUI. The students will complete four short activities and answer a post-test followed by an attitudinal survey to assess their attitudes towards computing and enjoyment surveys to assess their feeling about the tool. The practical session should last approximately 30 to 40 minutes.

For part 2, I will interview the children individually and asked some questions related to programming concepts that they have learned in the previous session. The interview will last around 20 minutes and will take place at School during normal hours.

All sessions will be video recorded using a camera positioned in such a way as to capture children’s spontaneous gestures and their facial expression to gather insight about their enjoyment.

**CONSENT**
Parents/carers will be sent an information letter with details of the study and a permission form to be signed and returned in order for their child(ren) to participate in the study. Parents/carers should be given at least one week to read and respond to this letter. Children will be asked for verbal consent to participate in the study after receiving an initial briefing on the nature of the study and the procedures involved. Children will also be given the option to withdraw from the study at any point.

Please sign below to confirm that you:

- Understand the requirements of children who take part in the research
- Have received detailed descriptions of the methods and materials to be used
- Give approval for the research to take place at your school

Name of school: _________________________________________
INFORMATION & CONSENT SHEET

PROJECT NAME

Interfaces and spontaneous gestures in primary programming education

INVITATION TO TAKE PART

Your child is invited to take part in a research study that aims to better understand the advantages and disadvantages of common approaches to teach programming to primary school students. The study consists of two parts. Part 1 aims to investigate the role of physical manipulation in learning programming. To do this we will compare a tangible user interface (TUI) that uses physical programming blocks and controls a physical robot, with a graphical user interface (GUI) that uses touch-screen programming blocks and controls an on-screen robot. Our objective is exploring the relationships between interface types, attitudes towards computing, enjoyment and gender. Part 2 aims to investigate children’s spontaneous gestures when explaining programming concepts. There is evidence for the importance of gestures in other fields of learning, but it is still emerging in the computing education field.

Thank you for carefully reading this information sheet, a copy of which you can keep for your records. This study is being conducted by student researcher Abrar Almjally from the School of Engineering and Informatics, University of Sussex, UK who is happy to be
contacted if you have any questions (a.almjally@sussex.ac.uk). The University of Sussex has insurance in place to cover its legal liabilities in respect of this study.

WHY HAS MY CHILD BEEN INVITED TO TAKE PART AND WHAT WILL THEY DO?
We are expecting to run the study with 50 children from year 1 of School. Part 1 should last approximately 30 to 40 minutes and will take place at School during normal hours. We aim to measure learning outcomes and attitudes by asking each child to complete a short pre-test and an attitudinal survey before the learning activities. In the session, your child will be assigned to work in pairs and given one-to-two instruction on one of the interfaces (either the TUI or the GUI) to program a robot, complete four short programming activities and answer a post-test and attitudinal survey to assess their attitudes towards computing and enjoyment surveys to assess their feelings towards the tool.

For part 2, I will interview the children individually and asked some questions related to programming concepts that they have learned in the previous session. The interview will last around 20 minutes and will take place at school during normal hours.

All sessions will be video recorded using a camera positioned in such a way as to capture children’s spontaneous gestures and their facial expression to gather insight about their enjoyment.

ARE THERE ANY RISKS TO TAKING PART?
During the session, your child will interact with physical and paper-based learning materials, all of which have been approved by the School. The learning activities will pose no additional risk to your child’s safety.

WHAT WILL HAPPEN TO THE RESULTS AND MY CHILD’S PERSONAL INFORMATION?
The results of this research may be written into a scientific report for an Informatics PhD thesis and/or publication. We anticipate being able to provide a summary of our findings on request from 01/04/2020 (a.almjally@sussex.ac.uk). Your child’s anonymity will be ensured in the way described in the consent information below. Please read this information carefully and then, if you wish for your child to take part, please complete the personal information questions and sign to show you have fully understood this sheet and that you consent for your child to take part in the study as it is described here.

CONSENT

- I understand that I am agreeing to allow my child to take part in the University of Sussex research described here and that I have explained the project to my child and I have read and understood this information sheet.
• I understand that my child’s participation is entirely voluntary, that they can choose not to participate in part or all of the study and that they can withdraw at any stage of testing without having to give a reason and without being penalised in any way.

• I understand I can request without penalty that my child’s data be withdrawn and deleted even after testing is complete, any time up until the results are analysed (01/1/2020).

• I consent to the processing of my child’s personal information for the purposes of this research study. I understand that such information will be treated as strictly confidential (subject to legal limitations) and handled in accordance with the General Data Protection Regulation (GDPR) 2016.

• I understand that my child’s collected data will be stored in a de-identified way (e.g. using ID numbers not names) and kept separate from other details about my child (e.g. from the consent form). Electronic data will be stored in a password-protected file and hard-copies will be stored in a locked drawer. De-identified data may be made publicly available through online data repositories or at the request of other researchers.

• I understand that my child’s identity will remain confidential in any written reports of this research and that no information I disclose will lead to the identification in those reports of any individual either by the researchers or by any other party, without first obtaining my written permission.

• I understand that my child’s name and data will not be shared with any third party outside the research group, unless I later provide written permission.

• I consent that the session will be audio/video recorded and understand that the recordings will be stored anonymously, using
password-protected software and will be used for specific research purposes.

- Please initial this box if you consent to the use of video material and images from your child’s participation in this study at academic conferences and research publications. (Note, Your child can participate in the study even if you do not initial this box)

- Please initial this box if you consent to share the video material and images from your child’s participation in this study with other interested researchers. (Note, Your child can participate in the study even if you do not initial this box)

Name of Child: _______________________________________

Name of Parent or Guardian: _______________________________________

Date: _______________________________________

Age of Child: _______________________________________

Signature: _______________________________________