tScratch: Tangible Programming Environment Targeted for Students who are Blind or Visually Impaired (BVIs)

DIANNE PAWLUK, VIRGINIA COMMONWEALTH UNIVERSITY (PI)
GIOVANNI FUSCO, THE SMITH KETTLEWELL EYE RESEARCH INSTITUTE (CO-PI)
Need

Problem: a lot of effort has gone into lowering the hurdles to programming for sighted students

- No such effort has been made for blind and visually impaired students (BVIs)
- **Worse!** The effort for sighted students (using graphical interaction) has eliminated opportunities for BVI students

Equal but separate is **NOT** the solution

- Most BVI students are in mainstream schools with sighted peers

Need: resources for BVIs to learn computer programming alongside their sighted peers

- Lower hurdles for BVI students will keeping them low for sighted students
- Improve engagement for BVI students while keeping engagement high for sighted students
Project Research Questions

Based on a comparison of the use of tScratch/Scratch versus a programming environment that uses a text based programming language:

Did the students in the tScratch/Scratch group improve more significantly from the intervention than the students in the text based programming group....

1. In their ability to apply programming concepts (concepts already specified)
2. In their ability to share in teamwork
3. In their perspective of working with visually impaired students
4. In their interest in computing science and STEM
5. In their view of their ability to have a career in computing science/STEM fields
6. In their view of inclusion in the education community
Outcome Assessment of Project

• One week computer camps
  o One with tScratch/Scratch, one with text based programming environment

• 6 BVIs and 6 sighted in each camp

• Assessment
  o Defined performance metrics for each student outcome
  o Defined assessments for each performance metric

• Pilot test 1 year before actual assessment
### Outcome Assessment of Project

<table>
<thead>
<tr>
<th>Student Outcome</th>
<th>Metric</th>
<th>Assessment Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Applying Programming Concepts</td>
<td>Understanding program someone else wrote</td>
<td>Pre/post Test, Student Portfolios</td>
</tr>
<tr>
<td></td>
<td>Write a program</td>
<td>Pre/post Test, Student Portfolios</td>
</tr>
<tr>
<td></td>
<td>Debug a program</td>
<td>Pre-post Test, Student Portfolios</td>
</tr>
<tr>
<td>Ability to Share in Teamwork</td>
<td>Share in program writing</td>
<td>Pre/post Group Projects, Peer Assessment</td>
</tr>
<tr>
<td></td>
<td>Contributed to programming</td>
<td>Pre/post Group Projects, Peer Assessment</td>
</tr>
<tr>
<td></td>
<td>Critique others</td>
<td>Pre/post Group Projects, Peer Assessment</td>
</tr>
<tr>
<td>Working with Students of Opposite Visual Condition</td>
<td>How treat students in the two visual condition groups</td>
<td>Pre/post Questionnaire, Assessment by Teachers in Pre/post Group Projects</td>
</tr>
<tr>
<td></td>
<td>View of students’ abilities to program in the two visual condition groups</td>
<td>Pre/post Questionnaire, Assessment by Teachers in Pre/post Group Projects</td>
</tr>
<tr>
<td></td>
<td>Comfort in working with students in the two visual condition groups</td>
<td>Pre/post Questionnaire, Assessment by Teachers in Pre/post Group Projects</td>
</tr>
<tr>
<td>Interest in CS/STEM</td>
<td>View of Computer Scientists</td>
<td>Pre/post Questionnaire</td>
</tr>
<tr>
<td></td>
<td>View of Computing Science/STEM</td>
<td>Pre/post Questionnaire</td>
</tr>
<tr>
<td>Career in CS/STEM</td>
<td>Frustration level in class</td>
<td>Assessment by Teachers Day 1 and Day 5</td>
</tr>
<tr>
<td></td>
<td>Sees obstacles to a career in CS/STEM</td>
<td>Pre/post Questionnaire</td>
</tr>
<tr>
<td>Inclusion in Education Community</td>
<td>View of how treated by teachers</td>
<td>Pre/post Questionnaire</td>
</tr>
<tr>
<td></td>
<td>View of how the two visual condition groups are treated by teachers</td>
<td>Pre/post Questionnaire</td>
</tr>
</tbody>
</table>
Tangible Environment Development

- Add a tangible + audio interface to the Scratch programming environment that can be used by BVI students alone or as part of a team of BVI and sighted students

- Break down the environment development into smaller concepts and focus on those that lower hurdles
  1. Tangible code block
  2. Tangible workspace
Code Blocks

1. Define Requirements and Constraints
   • physical connections of code blocks prevent invalid syntax
   • block assemblies indicate nesting
   • Identification of block category and block name
   • Etc.

2. Design Tables
   • Alternate designs and whether satisfy design requirements/constraints

3. Preliminary Physical Prototypes of Some Alternate Designs
   • Focus on arithmetic, relational and logical operators
3. Preliminary Physical Prototypes of Some Alternate Designs
   • Example: nesting (slots for operands can expand to contain other operator expressions)
   • Retractable cord/band
   • Telescoping tubing

4. Functional testing/robustness
   • Does it work the way we want
   • Is it repeatable/reliable

5. User testing
   • Compare most promising design versus controls (e.g., simple blocks, text based environment)
   • Users will be required to construct expressions involving operators, and read and edit expressions created by another person. Some expressions will require nesting.
   • Use metrics of correctness of task and completion time, questionnaire about experience
**Tangible Workspace**

1. Define Requirements and Constraints
   - Grouping of blocks into a script allows it be understood to go together
   - Moving of a group of code lines as a whole keeps things conceptually together (chunking)
   - Need to find/remember code blocks in code
   - Etc

2. Design Tables
   - Alternate designs and whether satisfy design requirements/constraints

3. Preliminary Physical Prototypes of Some Alternate Designs
   - Open workspace, event code blocks with suction cups, blocks connected by magnets
   - Gridded workspace that code blocks can slide along, blocks connected by magnets
3. Preliminary Physical Prototypes of Some Alternate Designs
   • Locating code
     • Grid system (either gridded workspace for code blocks or incised markings but still free movement)
     • Hand guidance (auditory or tactile feedback)

4. Functional testing/robustness
   • Does it work the way we want
   • Is it repeatable/reliable

5. User testing
   • Compare most promising designs versus each other and control (e.g., alternatives mentioned in a crossed design, text based environment)
   • Users will be required to construct expressions involving move code blocks, read and edit expressions created by another person, find a code block based on another person pointing it out.
   • Use metrics of correctness of task and completion time, questionnaire