Piaget and Programming Robots: Cognitive Developmental Level as a Predictor of Programming Achievement

Louise Flannery and Marina Bers
Piaget and Programming Robots

Cognitive Developmental Level as a Predictor of Programming Achievement
Tufts University
Eliot-Pearson Dept. of Child Development

DevTech Research Group

Learning, Thinking & Cognitive Development + Technology Design → Curricula & Classroom Implementation

Louise P. Flannery & Marina U. Bers

EETC

March 14, 2012
How can new technologies positively support children’s learning and development?
Big Ideas for Small Children

Louise P. Flannery & Marina U. Bers
EETC
March 14, 2012
Why Programming in Early Childhood?

- Revolutionary and yet not
  - Children creatively constructing artifacts is no modern trend!

- Foster high-level cognition
  - Computational Thinking - Analysis, decomposition, abstraction, algorithms
  - Constructionism - Build it, and the learning will come.

- Relationships with technology
  - Help children see themselves as active shapers of technological tools rather than passive consumers

- 21st Century Skills - Technological fluency, new media literacies, creative design / problem solving

- The technocentric trap: Tools are not inherently good/bad
  - Learning contexts matter at least as much as technological affordances
TangibleK In-Depth Study

- How do kindergarteners understand core concepts from the domains of programming and robotics?
- What is their learning trajectory towards these concepts?
Observations

**Correspondence Score Frequency**

- 10 children
- 19 children

**Completeness Score Frequency**

- 12 children
- 17 children
Cognitive Development from 4 to 6

Thought Patterns ~ Age 4 (Pre- / Intuitive operations)
- Elaborating & applying symbol systems
- Intuitive & transductive reasoning
- Focus on single perspective or feature

Thought Patterns ~ Age 6+ (Concrete operations)
- Systematic, more adult-like logic
- Use of empirical feedback
- Plan flexibly toward a goal
- Hierarchies & multiple classifications
Primary Research Question

How does cognitive development influence young children’s programming of a robot?

What are the implications for designing and teaching with technology in early childhood?
Sample

- 29 preschoolers and kindergarteners from Boston-area public and private schools
- 4.4 to 6.6 years old (5.6 on average)
- ~40% girls, 60% boys
- ~Half from urban neighborhoods, half suburban
- Nearly all had 1+ parent with a graduate degree
- 75% used a computer at home
- ~One third had experience with programmable robots
  - but it was unclear if the children had programmed the robots
# Measures – Programming Achievement

## Correspondence
- Selection of instructions for relevance to goal
- Scale from 0-5
- Based on support needed to achieve

## Final Program Completeness
- Correspondence + sequencing together
- Scale from 0-4
- Score based on how close the final program was to a complete solution
Cognitive Developmental Framework

Feldman’s Revised Piagetian Stages and Transitions

- Application of symbol systems
- Intuitive or transductive reasoning
- Gradual stage transitions
- Inconsistent use of new cognitive structures
- Logical reasoning
- Empirical observations
- Awareness of multiple dimensions leads to classification & hierarchies

Approximate age

2  4  6  8  10  12

Intuitive (Pre-) Operations  Concrete Operations

Exploration  Elaboration, Application  Exploration  Elaboration, Application

Louise P. Flannery & Marina U. Bers  EETC  March 14, 2012
## Measures – Cognitive Development

<table>
<thead>
<tr>
<th></th>
<th>Pre-Operational</th>
<th>Transitional</th>
<th>Concrete Operational</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal Focus</strong></td>
<td>Open exploration</td>
<td>Mixed</td>
<td>Given goal</td>
</tr>
<tr>
<td><strong>Initial Strategy</strong></td>
<td>None/intuitive</td>
<td>Intuitive with some systematic logic</td>
<td>Step-by-step</td>
</tr>
<tr>
<td><strong>Debugging Interest</strong></td>
<td>Indifferent to incorrect solutions</td>
<td>Interested but may not know how</td>
<td>Driven to find the best answer</td>
</tr>
<tr>
<td><strong>Debugging Strategy</strong></td>
<td>None/intuitive (guess-and-check)</td>
<td>Mix of intuitive and limited logical approaches</td>
<td>Logical &amp; flexible approach. Uses empirical feedback.</td>
</tr>
</tbody>
</table>
Results: Developmental Levels

- 48% 5.9 years old
- 28% 5.1 years old
- 24% 5.6 years old
Programming: Late Pre-operations

- **Focus**
  - Open-ended exploring or
  - Short Hokey-Pokey attempts

- **Strategies**
  - Identified 1-2 actions
  - Started over rather than revising a program
  - Moved on to other explorations – programming, Legos, etc.

- **End Results**
  - Called exploratory programs successful Hokey-Pokeys
  - Not much concern over unfinished task
Programming in the Transition

- **Focus**
  - Interested in the Hokey-Pokey challenge
  - Happily switches to other open explorations though

- **Strategies**
  - Some systematic progress or use of testing
  - Some intuitive or guess-and-check strategies

- **End Results**
  - Prototypes – e.g. the systematic revisions had been applied to making the program match in length rather than elements, or
  - Partially / nearly complete programs

Focus
- Some explored first, others started Hokey-Pokey right away
- After starting the Hokey-Pokey, they stayed on task until the solution was complete

Strategies
- Systematic reasoning: recalled song line-by-line
- Revision based on watching the robot
- Some also ‘read through’ their programs
- Recognized what was wrong and how to fix it

End Results
- Intent on getting the program exactly right
- Only 3 of the 14 children had one error left in the end
Comparison of Final Programs

- Late Pre-operations

- Transitional

- Early Concrete Operations
Results: Development and Achievement

Mean Correspondence Score by Cognitive Developmental Level

- Pre-Operational: 1.9
- Transitional: 3.9
- Concrete Operational: 5.0

Mean Program Completeness Score by Cognitive Developmental Level

- Pre-Operational: 0.1
- Transitional: 1.9
- Concrete Operational: 3.8

* Statistically significant, $p < .05$ and $p < .001$, respectively.

CD predicts 64% of correspondence variation

CD predicts 87% of completeness variation

Statistically significant, $p < .001$, for all comparisons.
# Implications for Curricula

## Pre-Operational
- Children can
  - Explore tools’ possibilities and limits
  - Work towards setting goals
- Teachers can support
  - Observation & reflection
  - Expand explorations
  - Model & scaffold systematic approaches to specific goals

## Concrete Operational
- Children can work on
  - Contextualized goals,
  - Open-ended exploration, especially for new concepts
- Teachers can support
  - Sharing of strategies
  - Expanding to increasingly complex goals and instructions
Implications for Design & Research

- Instruction set
  - Compelling across ages for introductory activities
- For older / more experienced CHERP programmers
  - More complex instruction sets
  - Lower-level instructions
- For younger children, embedded debugging support?
  - Or, is this the role of the teacher and child?
Future Directions

- Strengthening the current study
  - New, truly correlational study under way using a separate cognitive measure – initial data support the current results
  - Assess multiple programming activities per child
  - Sample size and characteristics

- Open questions to investigate
  - Compare achievement on tasks w/ actions only vs. control flow
  - Analyze the evolution of achievement over multiple activities
  - Research the process of moving from small, high-level instruction set to increasingly large, low-level set
  - Test curricular recommendations
Final Thoughts

- Developmentally-based trajectory for programming
  - Similar to other skills like drawing and block-building

- Reasoning skills for tailoring instruction as early childhood education includes more academic content
  - Teachers could estimate cognitive development / reasoning from existing activities

- Technology in early childhood
  - Scaffold rather than put off cognitively rich technology-based experiences like programming robots
  - Redefinition of ‘screen-time’
Acknowledgements

To Marina Bers, David Henry Feldman & Baktiar Mikhak, all of whose research and advising were invaluable to me throughout this project.

The DevTech Research Group, particularly the TangibleK team.
Thank you for coming!
References