TEACHING STEM, INCLUDING CODING AND ROBOTICS, TO STUDENTS WITH ASD

PRESENTED BY: DR. VICTORIA KNIGHT

THIS PRESENTATION WILL:

- Articulate the rationale for including students with ASD in STEM (science, technology, engineering, mathematics) curricula and contexts
- Focus on innovative research in this area to support this rationale, especially in the areas of robotics and coding
- Discuss methods I have used internationally to train content-based learning for all students in the classroom

WHAT IS INQUIRY-BASED SCIENCE?

- Diverse ways in which scientists study the natural world and propose explanations based on the evidence
- Activities through which students develop knowledge and understanding of scientific ideas
  - (NSTA position statement, 2015; Spooner, Knight et al., 2013)

WHY STRUCTURED INQUIRY?

(Knight et al., 2017)

OKAY, MAYBE EXPERIMENTS, BUT CODING? ROBOTICS?

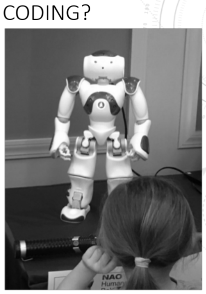


Image on left from: http://web.media.mit.edu/~mres/papers/Scratch-CACM-final.pdf; image on right from: https://www.roboticsbusinessreview.com/consumer/robots-help-teach-stem-to-students-with-autism/

ROBOTICS STUDIES

(KNIGHT, WRIGHT, & RUPPAR; WRIGHT, KNIGHT, WILSON, & BUCHANNON; KNIGHT, WRIGHT, BUCHANNON, & WRIGHT)

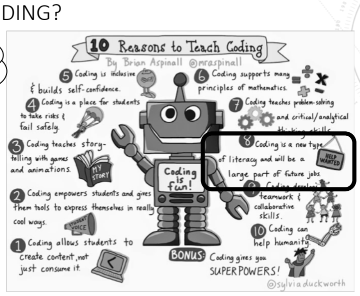
MATERIALS

INTRODUCTION

- What do we mean by "coding"?
  - Paper and marker version
  - Digital literacy version

WHY TEACH CODING?

Remember "shared interests?"



Wei et al. (2015); Image from:http://brianaspinall.com/10-reasons-to-teach-coding-sketchnote-by-sylviaduckworth/

SHARED LEARNING: TEACHING ROBOTICS AND CODING TO STUDENTS WITH ASD AND THEIR PEERS

KNIGHT, WRIGHT, & RUPPAR, IN PREPARATION

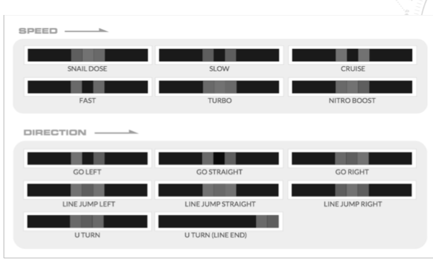
(Knight, Wright, & Ruppar, in preparation)

PURPOSE

- Gather descriptive data to determine facilitators and barriers to teaching coding of robots to children with ASD and typical peers
- Examine communication and engagement during high interest activities
- Explore collaboration and shared learning experiences between children with ASD and typical peers

(Knight, Wright, & Ruppar, in preparation)

SAMPLE CODES



(Knight, Wright, & Ruppar, in preparation)

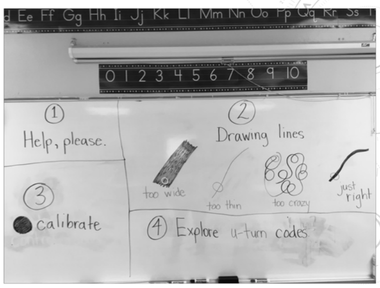
SETTING

- Summer Camp
  - Inclusive school in Central Texas
  - Structured program for students with autism
  - BCBA and RBTs as support staff
- 2 weeks
- 4-5 students with autism and 4-5 typical peers

(Knight, Wright, & Ruppar, in preparation)

LESSON STRUCTURE

1. Mini-lesson
2. Guided practice
3. Independent practice

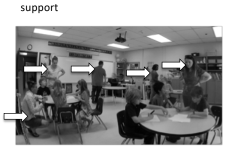


(Knight, Wright, & Ruppar, in preparation)

Example:Mini-lesson content

LESSON STRUCTURE

Guided Practice with para-educator support



(Knight, Wright, & Ruppar, in preparation)

Lesson on calibration & preparing the robot for coding.

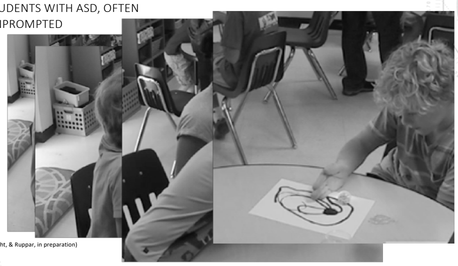
LESSON STRUCTURE

INDEPENDENT PRACTICE

(Knight, Wright, & Ruppar, in preparation)

PRELIMINARY FINDINGS

PEERS ORGANICALLY SUPPORTED STUDENTS WITH ASD, OFTEN UNPROMPTED



(Knight, Wright, & Ruppar, in preparation)

PRELIMINARY FINDINGS

OBSERVATIONAL LEARNING OCCURRED FROM TARGET STUDENT TO PEER

(Knight, Wright, & Ruppar, in preparation)

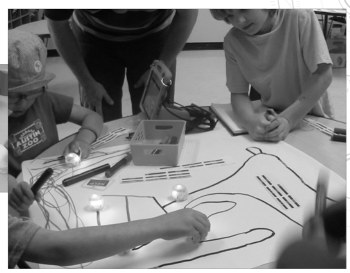
PRELIMINARY FINDINGS

OBSERVATIONAL LEARNING ALSO OCCURRED FROM PEER TO TARGET STUDENT

(Knight, Wright, & Ruppar, in preparation)

PRELIMINARY FINDINGS

MAKING CONNECTIONS: THE LEARNING SPACE MATTERS



(Knight, Wright, & Ruppar, in preparation)

PRELIMINARY FINDINGS

INDEPENDENT CODING OCCURRED

(Knight, Wright, & Ruppar, in preparation)

FACILITATORS & BARRIERS

- <u>Facilitators</u>
  - ➢ Same expectations
  - ➢ High-interest materials
  - ➢ Inquiry based instruction
  - ➢ Hands-on materials
  - ➢ Inclusive practices
  - ➢ Shared space
  - ➢ Lack of ownership of materials
  - ➢ Persistent peers
- <u>Barriers</u>
  - ➢ Social initiations
  - ➢ Motor skills
  - ➢ Intrusive prompting
  - ➢ Materials that promote isolation
  - ➢ Untrained peers & paras who help by doing

(Knight, Wright, & Ruppar, in preparation)

TEACHING CODING SKILLS TO HIGH SCHOOL STUDENTS WITH ASD AND EBD (USING THE DIGITAL VERSION)

- **Participants**
  - 3 students; 15-17 years old
  - ASD as primary diagnosis & EBD as an additional diagnosis
    - 1 student has selective mutism
    - 1 has severe/challenging behavior
    - 1 several phobias (including paper)
- **Setting**
  - Self contained school for students with EBD and other disabilities
  - High need urban school district

(Wright, Knight, Wilson, & Buchannon, in progress)

ROBOTICS: DIGITAL VERSION

ROBOTICS: DIGITAL VERSION (CONT.)

TEACHING CODING SKILLS TO HIGH SCHOOL STUDENTS WITH ASD AND CHALLENGING BEHAVIOR

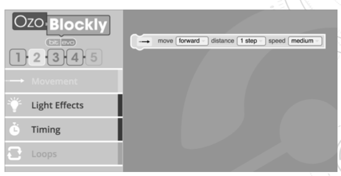
Design

Multiple probe across three participants

Single-case research design

Independent Variable

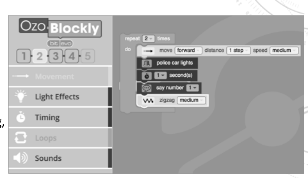
Explicit instruction: model-lead-test of one code (i.e., *movement*)



(Wright, Knight, Wilson, & Buchannon, in progress)

TEACHING CODING SKILLS TO HIGH SCHOOL STUDENTS WITH ASD AND EBD

- **Dependent variables**
  1. *Movement* code
  2. Generalization from *movement* code to a novel exemplar
  3. Generativity to novel exemplars (student self-selection, programming, and evaluation)



(Wright, Knight, Wilson, & Buchannon, in progress)

TEACHING CODING SKILLS TO HIGH SCHOOL STUDENTS WITH ASD AND EBD

- **Method**
  - Conducted a minimum of three baseline sessions
    - Measured the three dependent variables using a multiple opportunity probe
  - Intervention: Teach the movement skill using explicit instruction (model-lead-test)
  - Evaluation of :
    - Acquisition of the explicitly-taught *movement* code
    - Whether students can generalize to a new code (without explicit instruction; randomized)
    - Whether students can self-select the code, program, and evaluate, including these skills:
      - Decide what they want the robot to do
      - Create their own 5 step code
      - Test out the code
      - Evaluate whether or not their codes worked

KEY

Open circles: Explicitly-taught *movement* code

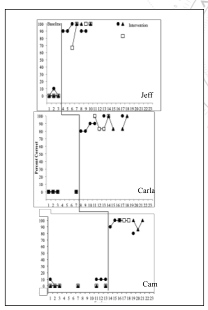
Triangles: Generalization to a novel exemplars

Square: Generativity of a student-directed code

RESULTS

IOA and PF for 30% of sessions, calculated at 100% IOA and 100% PF

(Wright, Knight, Wilson, & Buchannon, in progress)



DISCUSSION

- Functional relation between MLT instruction and student acquisition of (a) the explicitly-taught movement code; (b) generalization to novel exemplars, and (c) Student-directed code.
- **Implications for Practice**
  - Teachers do not need to teach every skill explicitly
  - There may be value in allowing students troubleshoot through trial and error for some tasks (high interest)
  - Students with ASD can engage in self-directed learning in STEM content
  - Students with ASD (many who typically lack resiliency) may be more inclined to try again when materials are motivating
  - All students played with robots once foundational skills were mastered

TEACHING CODING TO ELEMENTARY STUDENTS WITH ASD AND INTELLECTUAL DISABILITY (ID)

- **Participants**
  - 4 students
    - Two in 3rd grade, Two in 4th grade
  - All have an ASD diagnoses
    - 3 are eligible for the AA-AAS
    - All have vocal verbal ability
- **Setting**
  - Self-contained classroom for students with moderate and severe disabilities
  - High need urban school district

(Knight, Wright, Buchanon, & Wilson, in progress)

TEACHING CODING TO ELEMENTARY STUDENTS WITH ASD AND ID IN SELF-CONTAINED SETTINGS

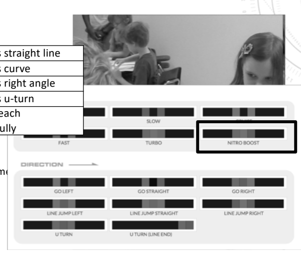
- **Design**
  - Multiple probe across skills single-case research design
- **Independent Variable**
  - Explicit instruction using model-lead-test of calibration, tracks, and one code

(Knight, Wright, Buchanon, & Wilson, in progress)

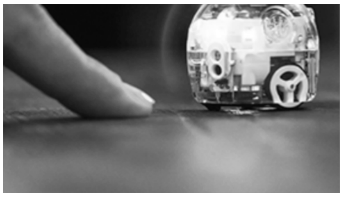
TEACHING CODING TO ELEMENTARY STUDENTS WITH ASD AND ID IN SELF-CONTAINED SETTINGS

- **Dependent variables**
  - Skill 1: Calibration
  - Skill 2: Drawing Tracks
  - Skill 3: Creating a code
- **Method**
  - Baseline
    - Teach vocabulary using constant time
  - IV
    - Teach skills using model-lead-test

1. Draws straight line
2. Draws curve
3. Draws right angle
4. Draws u-turn
5. Tests each successfully



(Knight, Wright, Buchanon, & Wilson, in progress)



RESULTS: PARTICIPANT #1

Tier 1: Calibration

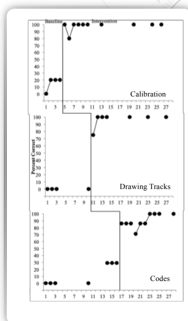
Tier 2: Drawing Tracks

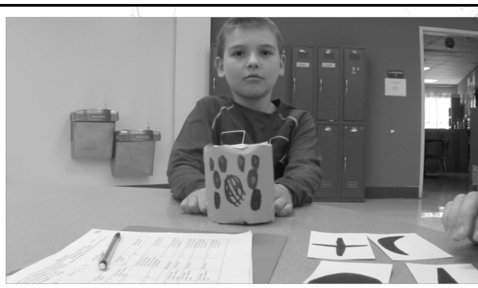
Tier 3: Codes

IOA and PF for 30% of sessions, calculated at 100% IOA and 98% PF

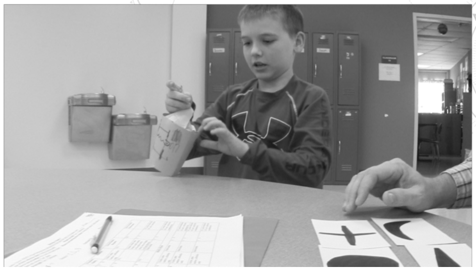
(Due to instructor waiting too long for the initiation of behavior)

(Knight, Wright, Buchanon, & Wilson, in progress)

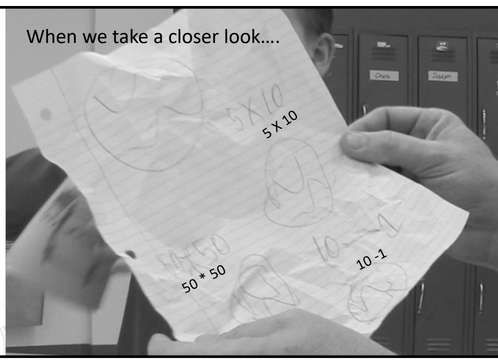




(Knight, Wright, Buchanon, & Wilson, in progress)



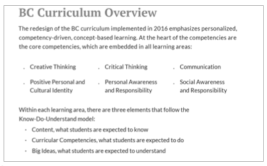
(Knight, Wright, Buchanon, & Wilson, in progress)



FUTURE DIRECTIONS

- The descriptive study highlights: joint attention, observations, observational learning, engineering skills (testing and re-testing), independent coding
  - Additional research is needed to examine these factors
- The dependent variable from the high school study suggests: generativity included components of resiliency, self-directed learning and problem-solving,
  - future research should attempt to parse out these outcomes
- Include classroom teachers as instructors in general education contexts
- Include peers in an inclusive setting
- Examine self-instruction as the independent variable (e.g., through video modeling)
- Evaluate collateral gains in teaching coding (e.g., communication, joint attention)
- Determine the role of preference assessments for instructional materials

CODING IS DIRECTLY TIED TO BC'S NEW CURRICULUM



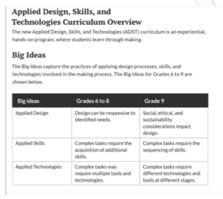
BC Curriculum Overview

The redesign of the BC curriculum implemented in 2016 emphasizes personalized, competency-driven, concept-based learning. At the heart of the competencies are the core competencies, which are embedded in all learning areas:

- Creative Thinking
- Critical Thinking
- Communication
- Positive Personal and Cultural Identity
- Personal Awareness and Responsibility
- Social Awareness and Responsibility

Within each learning area, there are three elements that follow the Know-Do-Understand model:

- Content, what students are expected to know
- Curricular Competencies, what students are expected to do
- Big Ideas, what students are expected to understand



Applied Design, Skills, and Technologies Curriculum Overview

The new Applied Design, Skills, and Technologies (ADST) curriculum is an experiential, hands-on program, where students learn through making.

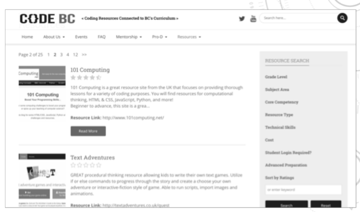
Big Ideas

The Big Ideas capture the practices of applying design processes, skills, and technologies involved in the making process. The Big Ideas for Grades 6 to 9 are shown below.

Big ideas	Grades 6 to 8	Grade 9
Applied Design	Design can be responsive to identified needs.	Social, ethical, and sustainability considerations impact design.
Applied Skills	Complex tasks require the acquisition of additional skills.	Complex tasks require the sequencing of skills.
Applied Technologies	Complex tasks may require multiple tools and technologies.	Complex tasks require different technologies and tools at different stages.

CODING RESOURCES CONNECTED TO BC'S CURRICULUM:

- http://codebc.ca/
- https://vimeo.com/188092764



ONE EXAMPLE: SCRATCH

- What is SCRATCH?
- Scratch is a free, graphical programming environment from MIT. It teaches 8- to 16-year-olds programming by snapping code blocks together to form complete programs.
- The site's collection of projects is wildly diverse, including video games, interactive newsletters, science simulations, virtual tours, birthday cards, animated dance contests, and interactive tutorials, all programmed in Scratch.

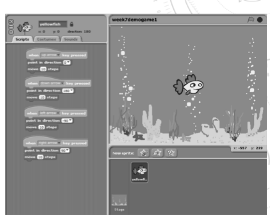


Image from: https://www.pinterest.ca/pin/168885673311151221/

SCRATCH

HOW TO: STEP by STEP

- • Go to https://scratch.mit.edu/projects/editor/?tip_bar=home
- • Click on "Create"
- • Click on the ? Icon
- • Click on "Step by step"
- • Click on "Getting started with SCRATCH" and "Begin"
- • To create, Click the X out of the tutorial
- • Use the Scripts, Costumes, and Sounds to drag and drop "blocks" of code into the programming space

Additional resources:

- SCRATCH ED:
  - http://scratched.media.mit.edu
- Invent with scratch: guides for teaching video games
  - https://inventwithscratch.com/
- Handbook for parents and teachers:
  - https://inventwithscratch.com/Scratch_Class_Handbook.pdf

SOME FREE CODING RESOURCES: ELEMENTARY STUDENTS

1. Blockly (for ages 8+)
https://blocklygames.
appspot.com/

• 2. Code Combat (for ages 5-17)
https://codecombat.com/

• 3. Code.org Studio (for ages 4-14)
https://studio.code.org/

• 4. Kodable (for ages 4-11)
https://www.kodable.com/

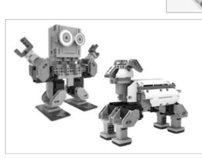
• 5. Scratch (for ages 8-16)
https://scratch.mit.edu/

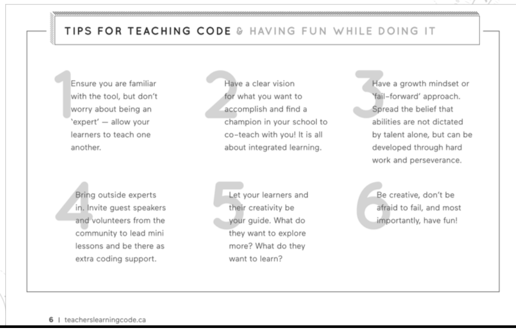
• 6. Tynker (for ages 4-14)
https://www.tynker.com/

SOME ROBOTICS RESOURCES:

Robotics kits that can be purchased:

- Cozmo, by Anki
- Bee-Bot and Pro-Bot,
by Terrapin Software
- Ozobot, by Evollve
- Buzzbots
- LEGO MINDSTORMS robots by LEGO





TIPS FOR TEACHING CODE & HAVING FUN WHILE DOING IT

1. Ensure you are familiar with the tool, but don't worry about being an 'expert' — allow your learners to teach one another.
2. Have a clear vision for what you want to accomplish and find a champion in your school to co-teach with you! It is all about integrated learning.
3. Have a growth mindset or 'fail-forward' approach. Spread the belief that abilities are not dictated by talent alone, but can be developed through hard work and perseverance.
4. Bring outside experts in. Invite guest speakers and volunteers from the community to lead mini lessons and be there as extra coding support.
5. Let your learners and their creativity be your guide. What do they want to explore more? What do they want to learn?
6. Be creative, don't be afraid to fail, and most importantly, have fun!

6 | teacherslearningcode.ca

QUESTIONS OR COMMENTS?