Teaching Programming on the Autism Spectrum: An Experience Report

Kurt Eiselt
University of British Columbia
Department of Computer Science
Vancouver, BC
Canada
eiselt@cs.ubc.ca

Jenny Sojat
ABLE Developmental Clinic
West Vancouver, BC
Canada
jennysojat@gmail.com

ABSTRACT
This paper describes an introductory computer programming course that combines technical education with social and communication skill training. Our goal is to provide youth with high-functioning autism both the technical and social skills that will facilitate their transitions to post-secondary education and the working world.

Categories and Subject Descriptors
K.3.2 [**Computers and Education**]: Computer and Information Science Education – *computer science education, curriculum.*

Keywords
Autism, computer programming, social skills, communication.

1. INTRODUCTION
In March 2014, the Centers for Disease Control and Prevention (CDC) issued a report with an alarming statistic: based on health and educational records of all eight-year-old children in 11 U.S. states in 2010, 1 in 68 children have autism [1]. Perhaps even more alarming is the fact that the estimate increased dramatically in relatively few years: in 2007, the CDC reported that 1 in 150 children were thought to have autism [2].

What is autism? "Autism spectrum disorder (ASD) is a range of complex neurodevelopment disorders, characterized by social impairments, communication difficulties, and restricted, repetitive, and stereotyped patterns of behavior....The hallmark feature of ASD is impaired social interaction. As early as infancy, a baby with ASD may be unresponsive to people or focus intently on one item to the exclusion of others for long periods of time. A child with ASD may appear to develop normally and then withdraw and become indifferent to social engagement." [3]

Many readers will be familiar with heartbreaking stories of parents of seemingly typical infants who suddenly become withdrawn and non-verbal. But as the word "spectrum" in "autism spectrum disorder" suggests, autism manifests itself in different ways. Some children with severe ASD show all symptoms in severity, while others with milder forms of ASD have subtle social deficits such as maintaining friendships and forming positive impressions with employers.

What is too-often overlooked in the on-going and growing discussion of children with autism and how best to help them is the fact that these children grow up to be adults with autism, with the same impaired social interaction abilities. What are the long-term prospects for these adults?

One might expect that children with high-functioning autism have a reasonably good chance of living adult lives comparable to their "neurotypical" peers. Unfortunately, their future isn't quite so promising. According to the CDC, nearly half of the children with ASD have average or above-average intelligence [4]. Furthermore, children with autism often exhibit abilities above their age level in niche areas such as non-verbal reasoning, reading, memory, and computing. [5] Nevertheless, as high-functioning ASD children become high-functioning ASD adults, they will often have difficulty finding and keeping jobs due to social impairments. In fact, a recent study indicates that people with a diagnosis of high-functioning autism[1], even those with average intelligence, are no more likely to find a spouse or hold down a job than those with severe forms of autism. [6]

In short, the long-term employment prospects for individuals with high-functioning autism appear to be no better than those for people with more disabling variants. Even those with average or above average intelligence, but unable to find or hold a job, are likely to depend upon the care of others for their entire lives. That care comes with a cost: a study funded by Autism Speaks indicates that the lifetime cost of providing care for someone with autism is $1.4 million (USD) when there is no intellectual disability, and $2.4 million when intellectual disability is involved. The yearly cost to the United States is estimated to be $236 billion. [7] Other studies report different dollar amounts [for example, 8], but in every case the costs are overwhelming.

Some have suggested computing as a vehicle for offering more promising futures to young people on the autism spectrum, the idea being that since some children on the spectrum demonstrate above-average abilities in computing [5], they may have futures, and possibly well-paid futures, in the computing industry. Recognizing the potential in the ASD community, the software

WCCCE'15, May 8, 2015, Nanaimo, British Columbia, Canada.

[1] [6] specifically targets people with a diagnosis of pervasive developmental disorder not otherwise specified (PDD-NOS), which at the time of publication was considered to be a distinct classification of autism spectrum disorder. Since the publication of the fifth edition of the *Diagnostic and Statistical Manual of Mental Disorders* (American Psychiatric Association, 2013), PDD-NOS and Asperger syndrome are no longer separate diagnostic categories, and most people previously diagnosed with PDD-NOS or Asperger syndrome are now included under a single autism spectrum disorder umbrella.

company SAP has announced its intention to hire people with autism for programming and testing. [9] Similarly, the nonPareil Institute provides technical training to adults with ASD with the expectation that its graduates will earn a living in computing. [10] Both efforts are commendable and the success stories are noteworthy. Nevertheless, there may be room for improvement.

In light of the issues raised in the preceding discussion, two concerns become immediately apparent. First, efforts such as those made by SAP and nonPareil focus primarily on technical ability. Second, these efforts could be applied earlier. The authors of this paper, a computer science instructor and a registered speech-language pathologist (SLP), set out to explore a merger of computing education and social skills education. We envisioned a series of computer programming classes in which, from the students' perspective, the technical component appeared to be the focus, yet we could create situations for the students to work on understanding and fulfilling social expectations. Our hope is that, over the long run, classes organized along these lines would provide our students with sufficient computing skills to enable potential employers to see past our students' diagnoses, while at the same time providing social and communication skills that would make it easier for our students to fit in the working world. However, before we could offer even pilot versions of these classes, we needed to select our first cohort of students.

2. THE STUDENTS

Nine teens and pre-teens were invited to attend the programming classes. All students had a diagnosis of autism spectrum disorder, and all were on the high-functioning end of the spectrum. All had expressed interest in computers or programming prior to the beginning of the classes, and all were able to bring their own laptop computers to class.

Eight students continued to the end, the ninth having dropped out early because he wanted more advanced content. Of those eight students, seven had attended weekly structured social groups sometime during the six months prior to the beginning of the programming class. These groups utilized the Social Behavior Mapping framework created by Michelle Garcia Winner [11] for improving individual social skills. Consequently, most of the students had been introduced to a common vocabulary for talking and thinking about social behavior, and this same vocabulary was used during the programming classes to cue desired social behavior.

3. THE PROGRAMMING COURSE

3.1 Curriculum

The course ran for eight weeks, with each class going for two hours per night, one night per week. We took a three-week break after the fifth class to accommodate external commitments on the part of the authors (e.g., conferences).

As we plotted the first few weeks of the curriculum, we went forward with a few assumptions in mind:

- Pictures might hold the attention of this particular group better than text, so we should start with a visual programming language instead of a text-based language. We chose Scratch.
- Everyone in this group plays computer games, so games might hold their attention better than anything else we can work on. We had them creating games as soon as we could.
- This group isn't keen on listening to someone talk to them for minutes at a time; they would rather be doing something. We adopted an informal "I do, we do, you do" model: the programming instructor talked and coded through a problem (the instructor's laptop display was projected on the wall), then the instructor and the students talked and coded through a related problem, and then the students worked their way through a third problem while the instructor walked among the students and provided assistance as needed.

The resulting course looked like this (many of our exercises were adapted from [12]):

Day 1: Topics: Drawing shapes. Moving the sprite.

Details: Introduce the motion, events, control, sound, and pen palettes; x- and y-coordinates; movement; degrees of rotation; first loops.

Homework: Create a Scratch script that writes your name on the screen.

Day 2: Topics: Create an aquarium.

Details: Build on motion, coordinates, loops; add looks and sensing palettes; introduce animation through movement plus changing costumes: sense edges and reverse direction; sense sprite-to-sprite contact and take action using if-then.

Homework: Add another animal to the aquarium that eats one of the existing animals.

Day 3: Topics: Pong

Details: The aquarium is just like a game, but it doesn't accept user input; Pong has the same features (game loop, collision detection) and adds user input; build on sensing, conditionals, motion, coordinates, loops; create a paddle that responds to key press to move the paddle.

Homework: Add second paddle for two-player Pong. Use variables to keep track of the scores.

Day 4: Topics: Telling a story through programming.

Details: Our client needs some animation to introduce herself at a conference talk; the animation must meet specific written requirements (it's biographical and it has a time limit); pairs of students work together on Scratch programs that tell the story of one segment of the client's life.

Homework: Finish what was started in class.

Day 5: Topics: Hoppy Cat to Flappy Bird.

Details: A "midterm exam" of sorts; given the "Hoppy Cat" script as a starting point (the Scratch cat sprite hops over oncoming hurdles when a key is pressed), create the "Flappy Bird" scripts (the bird sprite navigates oncoming gates via key presses); work in pairs while adapting the existing script to the new task. (See Figures 1 and 2.)

Homework: Finish what was started in class.

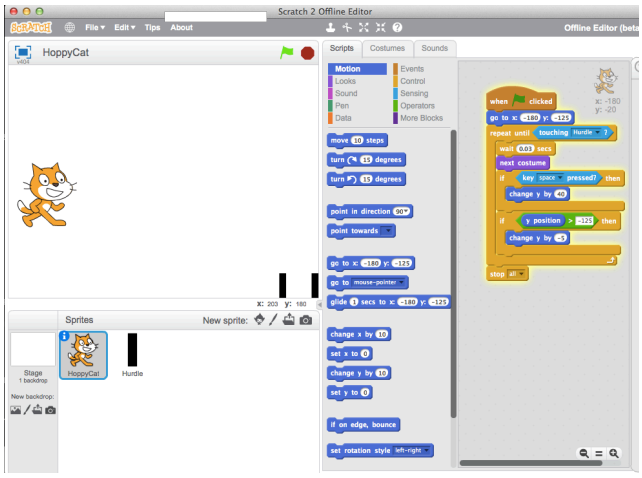

Figure 1. Hoppy Cat in Scratch

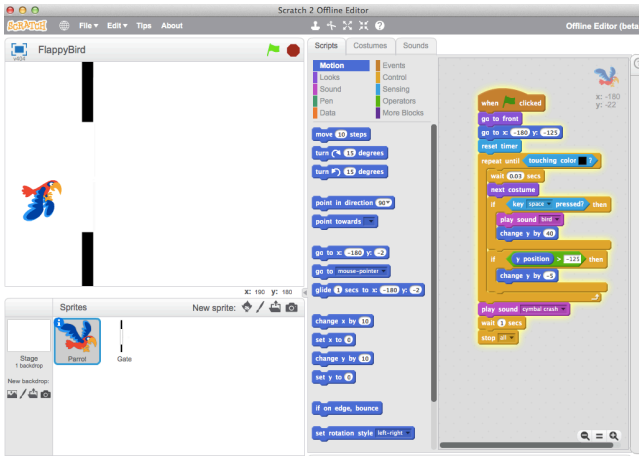

Figure 2. Flappy Bird in Scratch

Day 6: Topics: Whack-a-Mole.

Details: Build on and reinforce ideas we have worked with previously; introduce time for keeping track of how long the mole will appear on the screen; use random numbers to vary where the mole will appear on the screen; use mouse to position the mallet and "whack" the mole.

Homework: Finish what was started in class.

Day 7: Topics: Space Invaders.

Details: Don't rewrite a program you've already written; adapt the Pong paddle to be the Space Invaders cannon; start small then scale up; begin with one space invader that does not move; add movement to the one space invader (moves left and right across the screen while gradually approaching the bottom of the screen); introduce the cloning of sprites.

Homework: Use clones to try to create a row of multiple space invaders moving in synchrony.

Day 8: Topics: More Space Invaders.

Details: Finish the homework; add multiple rows of space invaders.

3.2 In-class Social Strategies

Ostensibly, the focus of these classes was computer programming. We were equally concerned with the social aspects of our classes, and we were especially interested in using the computer programming classroom as a venue for our students to practice the social skills training they had received prior to the course (and, in many cases, continued to receive in parallel with the course). Specific social challenges we either anticipated or observed early on included:

- Following group instructions, especially once laptops were open and programming had begun.
- Working in groups, taking feedback from peers, giving feedback to peers, initiating comments or questions, working on a common goal, and sharing responsibilities as part of a group.
- Taking on the perspective of others (knowing what others are thinking about you, what kind of impression you're giving, and why any of this matters).
- Knowing how and when to ask for help.

To address these challenges, we implemented several classroom strategies to bring social thinking to the forefront on occasion:

- With laptops closed, we began every class with a review of social expectations (e.g., look at the instructor, not your keyboard, when the instructor is talking), before there was any discussion of programming. Reviewing the social expectations prior to the programming exercises allowed us to use the vocabulary of the Social Behavior Mapping framework to remind the students of our social expectations with small verbal (or sometimes visual) cues as needed during class.
- We used the Social Thinking Vocabulary [13][14] to address any social difficulties that came up, helping to carry over learned concepts from previous social groups the kids had attended.
- We promoted the idea that computer programming is often a collaborative effort by frequently employing exercises that required working together, giving insight to others, accepting feedback, and acknowledging the work of others. We gave direct instruction and feedback on how to do this during times when it was expected that the kids were working together.
- We used "Pause" moments to have the students stop whatever they were doing and reflect on whether or not they were meeting social expectations (e.g, Were you thinking about what the instructor was saying? Were you talking to anyone about your ideas?). These moments allowed for reflection by the whole group without having to single out one student who was not meeting social expectations.
- Toward the beginning of every session except the first one, the students showed off the results of their homework efforts to the group. Toward the end of every session, we again held a "programming parade" in which each student demonstrated the day's accomplishments to their peers and, in some cases, their parents who had come to take them home. The parades provided the students with a structured time to ask questions and accept feedback. Creating separate events like these during the classes also allowed us to change social expectations: these were times it was

expected that students would *not* be thinking about their own projects and instead would be thinking about others.

4. OBSERVATIONS

Do we know that our eight students are "better" in one or more measurable ways? This is an experience report, which is a polite way of saying that we put in a great deal of effort, but at this time we have no real evidence that our students have changed for the better. At this time, we don't even know what exactly we might measure, or how. We'll discuss this a bit more in the next section.

We can, however, offer some purely anecdotal observations that may, at the very least, encourage others to give some thought to the issues outlined in this paper.

We think it's safe to say that many of our students know more about computer programming than they did when they started the course. Many of our students had little to no programming experience when they began the course, but by the end of the course they were able to read and write Scratch scripts (with varying degrees of confidence), and they were conversant in fundamental programming concepts such as loops, conditionals, and variables.

As for social interaction, we believe that we saw several improvements, but again, this is purely anecdotal. For example, we believe we observed an increase in spontaneous greetings to peers as they arrived for classes. We believe there was an increase in the students' ability to look at the work of others and offer positive and constructive comments, and that they were better able to receive feedback without taking it personally. We believe that, by the end of the eight classes, they were better able to maintain conversations with their peers, especially when the conversations were about programming.

One principle we stressed throughout the course was "Programming to Specifications": if you're writing a program for someone else, then write the program that was asked for, which may be different from the program that excites you at the moment. This might be obvious to neurotypical programming students, and it's a key principle in real-world software development, but our students found this to be quite challenging. People with ASD are thought to have difficulty seeing things from another person's perspective, or to imagine the thoughts and feelings of others. [15] Our students demonstrated a tendency to co-opt a programming exercise that we specified for them and make modifications that they wanted, without considering that they weren't fulfilling our expectations. The dilemma for us was to impress upon them the need to program to specifications without putting a damper on their desire to explore and experiment. Over time we learned to put specific boundaries around which components of the program were open to experimentation and which were not (i.e., "the space invaders may be anything you want them to be" versus "the space invaders must all move in the same direction at the same time"). We believe they now understand programming to specifications better than they did at the beginning of the course, but there is still much work to be done in that regard.

5. GOING FORWARD

Our intention is to provide more advanced classes for our original group of students while starting new students in the Scratch-based class. In fact, we are currently working with five of the eight students who completed the course in a course that combines social training with Java programming. The remaining three students wanted to continue, but we had changed locations and the parents of the students did not want to drive to the new location. Although several months passed between the end of the first course and the beginning of the second, the five continuing students have adapted to the new programming environment, but that's the topic for a future paper.

We are now working with an autism researcher from UBC's Faculty of Education to determine what, if any, assessments could be applied to future groups of students in order to measure improvements in social skills. When we know what we want to measure and are able to measure, we will start fresh with new students, track them through our computer programming and social training curriculum, and report the results.

6. CONCLUSION

Many individuals on the high-functioning end of the autism spectrum face a lifetime of assisted-care with little independence and erratic employment. The cost to parents, other care givers, and governments for providing care is astronomical and growing. Computing education may offer a way to a more fulfilling life as well as reduced care costs, but without complementary training in social and communication skills, the technical education may be wasted. We have taken the first small steps toward combining computer programming and social training in hopes of creating more and better life opportunities for this community, and look forward to refining and expanding our efforts.

7. ACKNOWLEDGMENTS

Our thanks to Pat Mirenda of UBC's Faculty of Education who is helping us with assessment, to Paul Carter of UBC's Department of Computer Science who is running our current Java-based course, to Claire Jones who is providing the social behavior cues in that same course, to Kathleen Delling-Eiselt who herds the cats, and to all our students.

8. REFERENCES

[1] CDC estimates 1 in 68 children has been identified with autism spectrum disorder. Retrieved February 18, 2015, from Centers for Disease Control and Prevention: http://www.cdc.gov/media/releases/2014/p0327-autism-spectrum-disorder.html

[2] CDC Releases New Data on Autism Spectrum Disorders (ASDs) from Multiple Communities in the United States. Retrieved February 18, 2015, from Centers for Disease Control and Prevention: http://www.cdc.gov/media/pressrel/2007/r070208.htm

[3] Autism Fact Sheet. Retrieved February 18, 2015, from National Institute of Neurological Disorders and Stroke: http://www.ninds.nih.gov/disorders/autism/detail_autism.htm

[4] 10 Things to Know About New Autism Data. Retrieved February 18, 2015, from Centers for Disease Control and Prevention: http://www.cdc.gov/features/dsautismdata/

[5] Characteristics. Retrieved February 18, 2015, from Autism Canada: http://autismcanada.org/aboutautism/characteristics.html

[6] Mordre, M., Groholt, B., Knudsen, A.K., Sponheim, E., Mykletun, A., and Myhre, A.M. Is Long-Term Prognosis for Pervasive Developmental Disorder Not Otherwise Specified Different from Prognosis for Autistic Disorder? Findings from a 30-Year Follow-Up Study. 2012. *Journal of Autism and Developmental Disorders*. 42 (June 2012), 920-928.

[7] Lifetime Costs of Autism Average $1.4 Million to $2.4 Million. Retrieved February 18, 2015, from Autism Speaks: https://www.autismspeaks.org/science/science-news/lifetime-costs-autism-average-millions

[8] Dudley, C., and Herbert Emery, J.C. The Value of Caregiver Time: Costs of Support and Care for Individuals Living with Autism Spectrum Disorder. 2014. *The School of Public Policy: SPP Research Papers*. 7 (January 2014), Issue 1. Retrieved February 18, 2015, from the University of Calgary: http://www.policyschool.ucalgary.ca/sites/default/files/research/emery-autism-costs.pdf

[9] Bryant, C. SAP seeks programmers with autism. May 21, 2013. Retrieved February 18, 2015, from Financial Times: http://www.ft.com/cms/s/0/2511f43a-c22b-11e2-8992-00144feab7de.html#axzz3S98xV7cJ

[10] nonPareil Institute: a non-profit technology company. Retrieved February 18, 2015, from nonPareil Institute: http://www.npitx.org

[11] Social Thinking. Retrieved February 18, 2015, from Social Thinking: https://www.socialthinking.com

[12] Ford, J.L. Jr. *Scratch 2.0 programming for teens* (2nd edition). Cengage Learning PTR, Boston, 2014.

[13] Winner, M.G., *Think Social! a social thinking curriculum for school-age students*. Think Social Publishing, 2008.

[14] Winner, M.G., *Thinking about you, thinking about me* (2nd edition). Think Social Publishing, 2007.

[15] Baron-Cohen, S. *Mindblindness: an essay on autism and theory of mind*. MIT Press, Cambridge, 1995.