Coaching an unending journey of discovery

Sastry S Duri, Feb 28th, 2021

I would like to share my perspective on coaching through couple of real examples while they are still fresh in my mind.

Example 1: On boarding of a member to FLL team

Ryan, not his real name, is a 7th grader. His mother reached out to me during first week of January. Here, I document the journey Ryan and I have taken since then. I thought this is the best way to communicate, as I was afraid the message will be lost in general statements.

Before my first meeting with Ryan I asked him to prepare a two-minute audio introducing himself, and play it during our first meeting. Ryan sent the recording a day earlier. I explicitly stated that he could record many times, and could send me whatever he feels best. I wanted to see if he makes multiple recordings or sends the first one he made. Makes no difference, but reveals a bit about the child.

[My assessment: he speaks well. Otherwise, it was what you would expect from a typical young boy.]

First meeting: Jan 10

Coach Q) how did you prepare and plan your talk? Ryan A) I wrote things up, and showed them to my mom and dad, and recorded it.

During the meeting he mentioned that he is familiar with Scratch, which is the programming language we will be using to program the robot this year.

I asked him if he is interested in developing a battleship game. He seems to know it. I gave the following description to him: The game consists of 10 X 10 grid, and has three elements: an aircraft carrier (4 units length), a destroyer (3 units in length), and a gun boat (2 units in length). Rest of the details all I left it to him telling him that it is his game so he should decide. I told him it would take 3-4 weeks to complete the exercise, and it may take more depending on what he wanted to do.

Coach Q) Why do you think I asked you to do write battleship game?
Ryan A) So you know how much I scratch I know?
Coach A) No.
Ryan A) To see how I think?
…
…
Coach A) this exercise helps us understand each other: how we think, how we communicate, and what words we use and what they mean.

I wanted to see how he deals with under specification. What reasons and logic does he use to fill in the gaps. Is he confident of taking those steps? Looks for direction? How does he ask for help? Defensive? Matter of fact? I asked parents not to help him.

I liked the fact he kept on giving answers to my question above, even after I rejected his earlier answers. My assessment is that he is undeterred about rejection, and very easy to work with.

Meeting: Jan 17th
Ryan shares the design of his game. He designed a two-person game. One person arranges the elements on the grid, then clicks a button to hide the arrangement of the elements. The other person uses grid coordinates to start bombing campaign. We talked about various design choices but did not suggest what he should do with them. One of the choices that Ryan made is to indicate failed attempt. I mentioned that others may not do so, as it makes game easy to play. I wanted to see what he would do. He stayed with his original choice.

End of the meeting I mentioned to that once he thinks the game development is correct, he should write a user manual on how to play the game. He needs to share that with my team. They will try to play the game and try to break his program, and give him feedback.

I told him not to aim to write a bug free program. Many programs written by humans have bugs. But the main purpose of the exercise is to develop skills to receive feedback, and receive it well. The team in turn will learn how to talk about failures respectfully and how to help others benefit from their feedback. So, it helps my team as well.

We had few more meetings. In these meetings, we tried to figure out how one person can play the battleship game as it is difficult to find a friend setting next to him. Final solution, computer randomizes the arrangement of the fleet. He learned about using random function in Scratch.

Meeting Feb. 7th

Making a robot turn 90 degree turn. We talked about how to program motors to make the robot turn. The drive train is made up of two wheels and a castor wheel.

Coach Q) Ryan, I know two ways in which you can make robot turn. Do you agree? If you know them can you explain how?
Ryan A) After 5-10 seconds thinking, he says there are three ways.

a) point turn (both wheels turn in opposite directions),
b) one wheel doesn't turn while the other wheel turns
c) both wheels turn

Ryan draws representative circles on paper. So, I know he knows what he is saying.
Coach explains what a wheel base is, and asks Ryan to make the robot turn 90 degrees. We talked about how wheel base and tire circumference are related. He drew some pictures of point turn and from that got the idea.

Then with coach's help, Ryan wrote a simple program that stops when light sensor sees black color. The task for next week was as follows: Ryan needs to figure out what is the thickness of the black line at which the which the robot fails to stop, that is, the light sensor fails. To create the black line Ryan could use black electrical tape.

My thoughts: sensors have limitations. You can't explain this to a kid. So, I ask them to see if they can make sensor fail. It is open ended. There is no one right answer. Lots of opportunities to explore, explain, and discuss.

Meeting Feb 14th

Ryan did not have black electrical tape. So, he created black line with black markers as shown below. I learned that this kid can improvise. Very good trait. In his experiments the robot failed to stop if the line less than 5 cm thick. This is not what I expected. In fact, I expected that he would come up with a line thinner than, say, 0.5 cm.

So, we tried to find out why that is. (From Ryan's perspective he accomplished what was asked of him: the robot stops when the line thickness is 5 cm. And his program "works". I did not challenge him.)

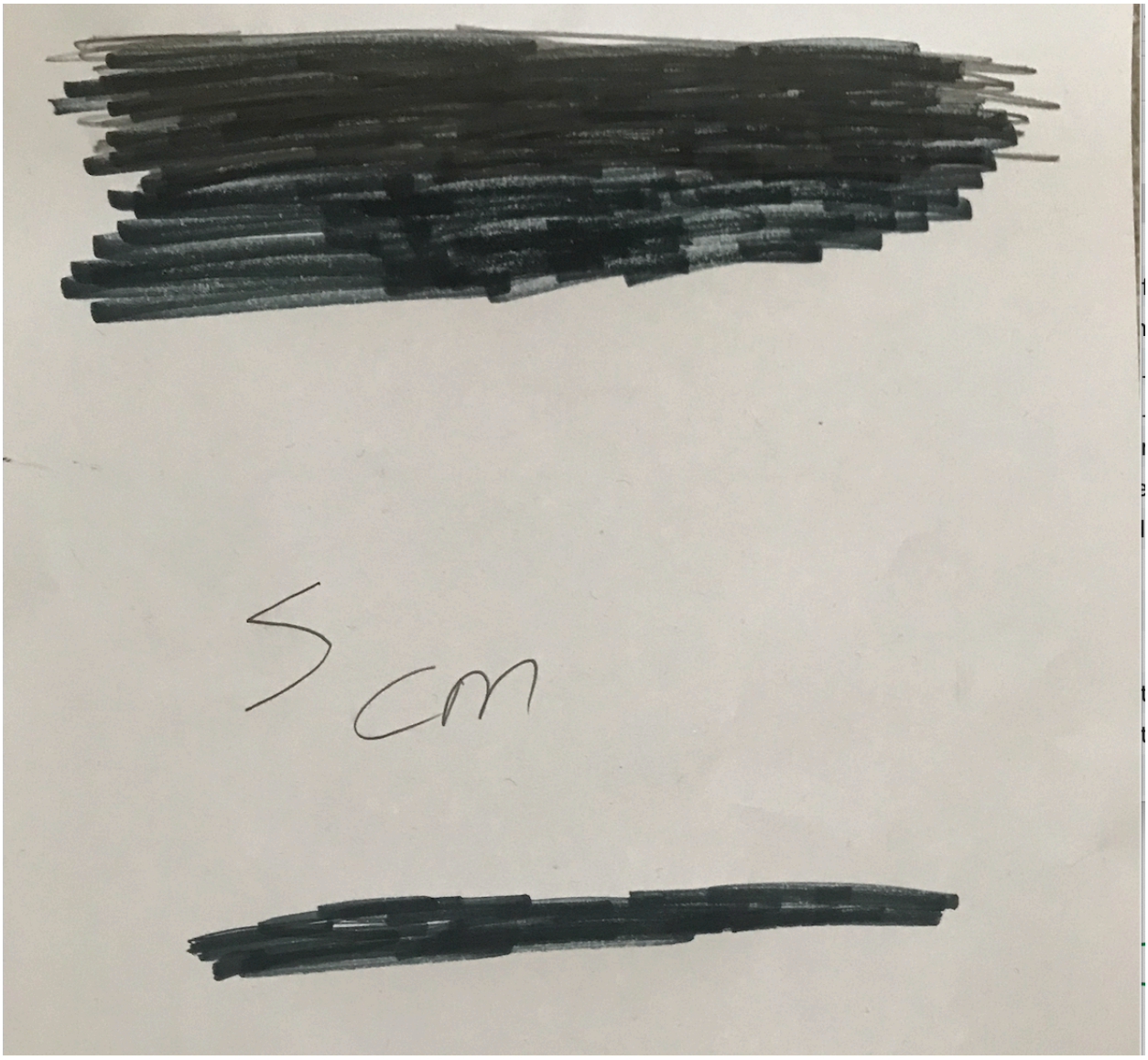

Figure 1 The test board Ryan prepared

I asked him to draw a thinner line, the one on the bottom and show me that the robot fails to stop. He did and the robot failed to stop. I noticed that the sensor is in “color” mode. I asked him to change it to “reflected light” mode. This time the robot stopped at the thinner line. The

robot stopped at the thicker line as well. We changed the sensor mode back to color mode, and it failed to stop at the thinner line. I asked him if he noticed anything, and asked him to summarize what we had found that day.

After some back and forth, he grasped the idea of color sensor mode. The next experiment continued in a similar way: make the light sensor fail.

For the following week, his task was to keep the robot in a black square of 15in and make robot go for 2 mins without stopping, while with in the square.

Meeting on Feb 21.
Ryan showed me the program which makes 90^0 predictable turns. I asked him to make a change so that when it hits the wall, it makes arbitrary turn. We talked about the random function and how he can vary the turn. We also talked about creating a black line with electrical tape and following the line.

Meeting on Feb. 28.
He showed the video of the robot making a random turn. My comment is we may not use this in competitions, but good to know how to use the random function. For next week, he plans to do the light sensor failing experiment, and line following experiment.

Example 2: How my team learned to program robot in scratch

Before this season my team used EV3 based robots and used Labview based visual programming language. This year, we started using SpikePrime. Three of my team members claimed prior programming experience, and one had no prior experience.

We started in April and continued till August.
Weeks 1-3:
Team is asked to write whatever Scratch programs they like and share that during weekend meetings. One wrote animated story. One wrote a game of laying bricks. One created a maze game. And another wrote program to calculate cos and sin values from given sides of a triangle. They refined these through for three weeks. This gave me an opportunity to assess their capabilities.

Weeks 3-8: Tick-Tack-Toe
We divided the team into pairs. Each pair meets during week as they see fit. And share their ideas during weekend reviews. And asked to write a tick-tack-toe game. Nothing more is given. I told them they may not be able to write complete game, but may learn a lot doing so.
Week4: One pair wrote person/computer game. Another pair wrote player vs player game. Neither game worked.

Week 5: Buggy. Terminal conditions had problems. They asked to review their programs with other pair: and they did by reading the code out loud. So, from week 6 onwards they are asked to present program design during code reviews.

Week 6: Team struggled to present their programs without referring to code. Programs worked most of the time. The morning of week 7, they were asked to share link to their game with other pair. The task for each pair was to break other pair's game, and find as many bugs as possible. And be prepare to give a gracious professional feedback without making the receiver of the feedback uncomfortable.

Week 7: Giving feedback was fun, and receiving it not as much. They practiced it ever since. Pairs were asked to implement other pair's high-level design—the goal here is to see how effective their design communication was.

Week 8: Pair implementing other pair's design reviewed code with them. There were surprises, and comments that the other pairs' design did not match with their own.

Weeks 9-12: **Battleship game**
During tick-tack-toe game I noticed that each pair developed very different code. But both pairs did write separate code block for each of the nine squares. So, I asked them write a Battleship game using 10X10 grid which makes it almost impossible to write per square program blocks. Also, the long conditional statements used to determine winner is no longer possible.

Without any introduction to lists or arrays both pairs naturally used them to keep track of state.

Helping them learn without teaching

The title of this section is very misleading, so let me explain it further. For example, when I noticed long conditionals used to determine the winner or per sprite blocks in tick-tack-toe game I could have taken some time to review and improve their code. But this would make me look smarter than my team, and would have diminished their accomplishments. I was thrilled with their capabilities. Instead, I gave them battleship game where such constructs are not possible or practical. The team again surprised me with their use of arrays and lists. To this day I have not closely reviewed any of their code so can't comment further. The code written by both pairs differed and that is good enough for me.

So, what did I teach? The concept of high-level design. The concept of testing and code breaking. The concept of giving feedback respectfully. I talk to them about agile design, fail fast approaches, setting up experiments, documenting and ways to evaluate different designs. How do you decide the current approach is failing and needs to be changed? How can you do it in a data driven way. Not everything sticks, but I try. Besides the technical skills we can develop character as well.

I followed similar approach with Ryan as well. I told him there is a function called random, and see if he could use it to improve his code. He read, and talked to my son to get some help. I was not part of help loop. Kids can help each other.

Role of a coach

As you see the journey of discovery is based on open ended exploration. I don't judge children on their abilities. I probe them to understand their traits to figure out how I can help them in their journey of discovery, and stay out of their way as much as I can. Early 90's I read a book called *Dibs In Search of Self*. My approach to coaching is much influenced by that book.

A coach, like a gardener, should prepare the land, plant seeds, and tend the garden patiently. The seeds sprout, the plants grow, and bloom according to their whim.

Why I am interested in how you profile?

In 2019, I visited a school in AP and spent one day with the children at the school. Hope was that I would train teaching/coaching staff remotely and they would run the program at the school. I found it difficult to communicate with teachers. I think the concept of open-ended exploration is alien to them. For example, I sent them about 7 experiments and told them it would take about 10 weeks to complete them. A week later they sent me videos containing the results of all experiments. From my perspective it was a disaster. But the teachers and students must have felt very accomplished.

My goal was to work with large set of children have back and forth discussions with children and analyze the words and patterns to learn when they understood the concepts. And use these learning to guide teachers become better coaches.