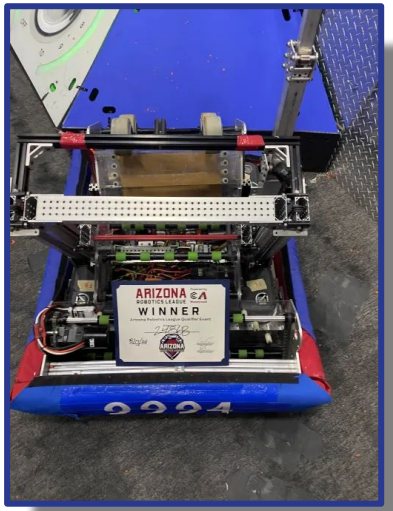
Offseason Projects: How To Achieve Sustainability

By: Eugene Ellsworth Bowers, Zane Guzman, Oscar Davis

Arizona Robotics League/ Offseason Competition

In order to achieve team sustainability we had the freshman at the time: Zane and Oscar along with many others, design build program and drive an entirely new robot for the Crescendo off season competition ARL. They would compete as a B team separate from the main 2486 team. This was the Coconuts way of achieving sustainability in our team, to help us flourish into future seasons when the older, and hence more experienced members, were no longer with us. In this presentation we will share personal experiences to try and show you how you can sustain your team through your own efforts as a new member, or through teaching as an experienced veteran.



How We Were Taught:

When learning our respective skills these were tactics that the older members used to teach use.

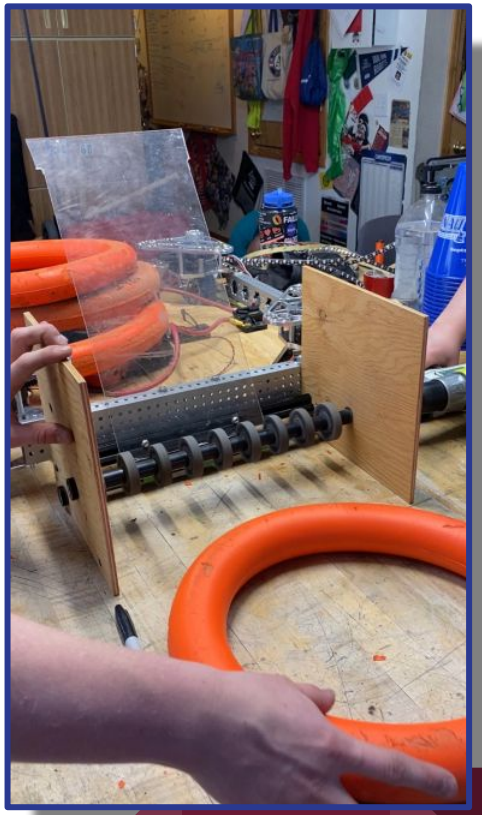
1. Give the skills, not the answers (giving the tools like how to cad gives a better basis for the future than designing the robot for them for example)
2. Help less as skills grow (as your new members skills flourish, your presence should diminish to give them a taste of the real experience)
3. Always answer questions in some way (the answer doesn't need to be directly given, but help should always be ministered. Imagine how you would feel in the situation!)

Cadding:

Designing your robot is one of the most crucial parts of the building process, as this is where your robot, and hence your goals, are formulated. We recommend making sure that you give your designer-designers a great basis of skills before you have them build a robot for ARL, or any other activity that you think would help with training. With CAD, from the perspective of a new designer, I found that there was a lot of information to take in at the start and I needed help. What helped me most of all was having an experienced member help me, and guide me through the process, as I took my first steps into designing this robot. Really my cadding training went through these easy steps:

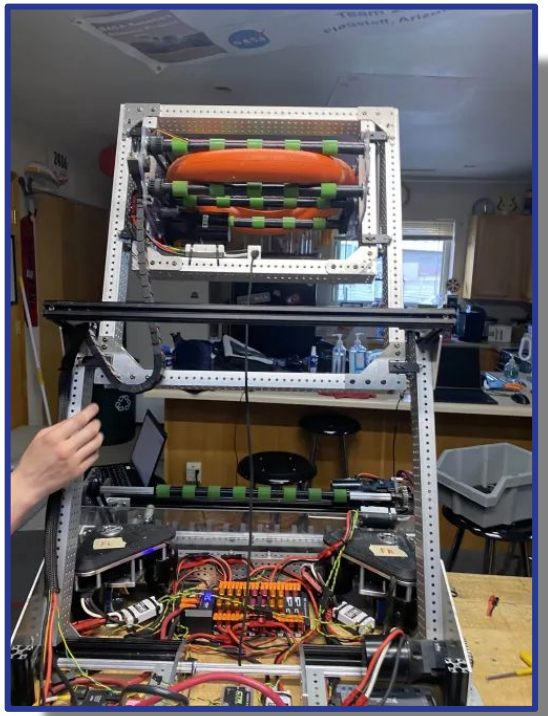
1. I was given the most basic skills of CAD (drawings and features. I was given this training in a schooling setting, but if you don't have that option, a higher experienced member should show you the ropes)
2. I was guided through the basics of how to CAD a robot
3. I was shown tips and tricks for how to do it better, faster, and more reliably
4. I was monitored, to make sure that I was CADing correctly and without problems
5. I was let loose to CAD on my own, but most importantly I was still allowed to ask questions.

These steps are what allowed me to learn all about CAD, while actually making a robot, and I think they show the basic steps for how someone should be trained.



Building:

Physically building the robot, is a much more year round thing to teach your disciples. You should have new members participate in smaller things to teach them the basics in the offseason before the main season, and during the main season. This is the best way to get hands on experience. Then once the season is over, we found that it was best to have the new learning members do their own bigger projects. Now, a robot has multiple parts, so you can escalate their involvement subsystem by subsystem. You can guide them through the building process of the first subsystem or piece, then slowly fade out of the picture as your future team takes over.



Programming:

Programming is one of the harder things to learn as a new member. This is because its harder to give new members the proper experience to truly develop programming skills because it takes a lot more time to learn the basics in the first place. First off you need to make sure they have a well founded understanding of the programming language your team uses, how I personally learned Java(the language we use) is through websites like w3schools.com and codecademy.com. Then the most important thing to do for them after that is to give as much time with programming a robot as possible, and to make sure to do it often because as a new programer it can be very easy to forget stuff fast.

Programming can be very hard and frustrating especially for a new member so be patient and don't give up with it.

I personally went barely any programming knowledge to enough to program an entire competition ready bot in a few months thanks to patient and very helpful mentors

Autos:

Making autos is one of the hardest parts of programming, mostly because of how long it takes and how tedious it is as it is a lot of trial and error. It should be one of the last things you teach a new programmer because of this, although there are ways to make the autos making process a lot easier. On the b-bot we used pathplanner for the first time and we learned as a team how great it can be for simplifying autos and making it easier for new members to make autos. Thanks to tools like this we were able to create a 4 note auto for the b-bot that worked pretty consistently.

Autos Contin

Driving:

Driving is a very important part of FIRST, and it is done by two people, the co-driver and the main driver. We recommend you find these people immediately by having them tryout using your main season robot. What we were tested on was

1. Maneuverability (how likely you are to hit things vs. not)
2. Cycle time (how fast you can do a task going from one side to the other)
3. Maneuverability+cycle time (these often go together)
4. Defensive capabilities (stopping another robot)
5. Offensive capabilities (scoring and dealing with another robot)

Then we found a co-driver that matched my driving style, and was good at COMMUNICATION

We were tested on these skills and we found who was the best fit to drive. Then because of my preferences we were able to easily have programmers assign buttons, and there was no extra deadline for ARL of choosing a driver.

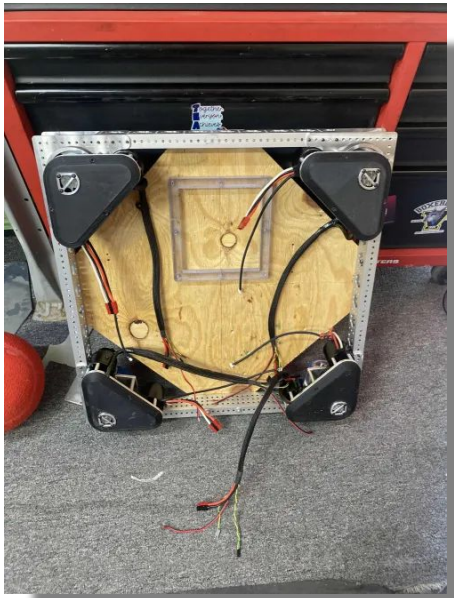
Quick note: I was once on an FTC team that waited until the day of competition to choose a driver. This usually leads to constant switching of people, and avoidable losses.

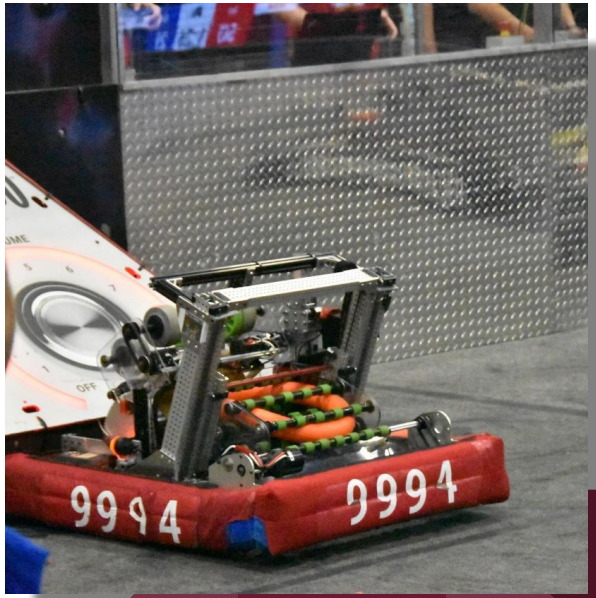
The Competition Experience:

Competitions are arguably the most stressful part of robotics, and administering experience and help during the competitions was how our team was able to solidify its sustainability. Here are the steps we followed talking the ARL approach.

1. Have your new members help during main season (pit crew, scouting, you don't need to overcrowded certain jobs, but you shouldn't leave places that could fit another person lacking)
2. During the B-teams first competition, help them as much as they need (its horrifying to be there for the first time. Our team tackled this by not even competing in one competition with the main bot, but if that's extreme, you can just have some dedicated helpers)
3. Slowly fade out of the competition picture, but if an issue ever arises, help as much as you can. Some situations you can't train for, you can only struggle through.

And that is how we went from a bunch of new members who knew nothing and a drive base to a bunch of members who are now teaching the new generation of coconuts and a competition ready robot





The Experiences We Gained:

- When working with the b bot I learned how to deal with things breaking and how to stay calm while fixing them, at our first competition when this first happened I had a really hard time not panicking but as the competitions went on I quickly learned how to stay calm and isolate the problem
- Throughout programming the b bot I had to learn how to troubleshoot and be patient with the bot no matter how hard it got. Often times it could take a whole day to troubleshoot just one bug.
- I learned how to fit requirements when building a robot, such as using a budget and present materials and time constraints.
- I learned how to work under pressure, and deal with losses.
- I learned how to work with people, and cooperate effectively

to complete a task.

Step 9994:

Make them make a presentation on team sustainability