Botball and Task Analysis!

Following is a supplemental document to accompany the "Hints for New Teams" document. It's a thorough explanation of how a team can use a "Task Analysis" approach to prepare for the Botball tournament.

This is just one of many approaches a team can take to plan and prepare for the Botball tournament. If you're struggling with where to start, give this a try! If you are successful with another technique we encourage you to share your results with us. We'd like to have a number of different successful techniques to share with new teams!

Game Strategy & Task Analysis

1. **Break the tasks up into smaller subtasks.** After the students understand the game and scoring possibilities they will immediately want to work on robots and strategies. Of course they will overestimate their capabilities and the time it will take them to complete a simple task. "I am going to build a robot that will do this (fly) and then this (incredible feat) and finish by doing this (swimming)", all complicated and possibly doable by Dr. Miller's grad students, but not your typical new Botball team. This is a teachable moment where you need to step in and facilitate by showing the students how to reach their overall goal by breaking it down into subtasks or "mini" tasks. (For teachers this is a Task Analysis - in order to do this the students must have this skill.)

   This has the benefit of making it easier for the students to complete and enjoy success on the way to completing the sequential tasks without being overwhelmed or distracted by the overall picture. If they complete a task analysis for all of the possible scoring solutions they generated for the current game, they can easily see what tasks are more complicated (have more steps) and what has to happen to accomplish them.

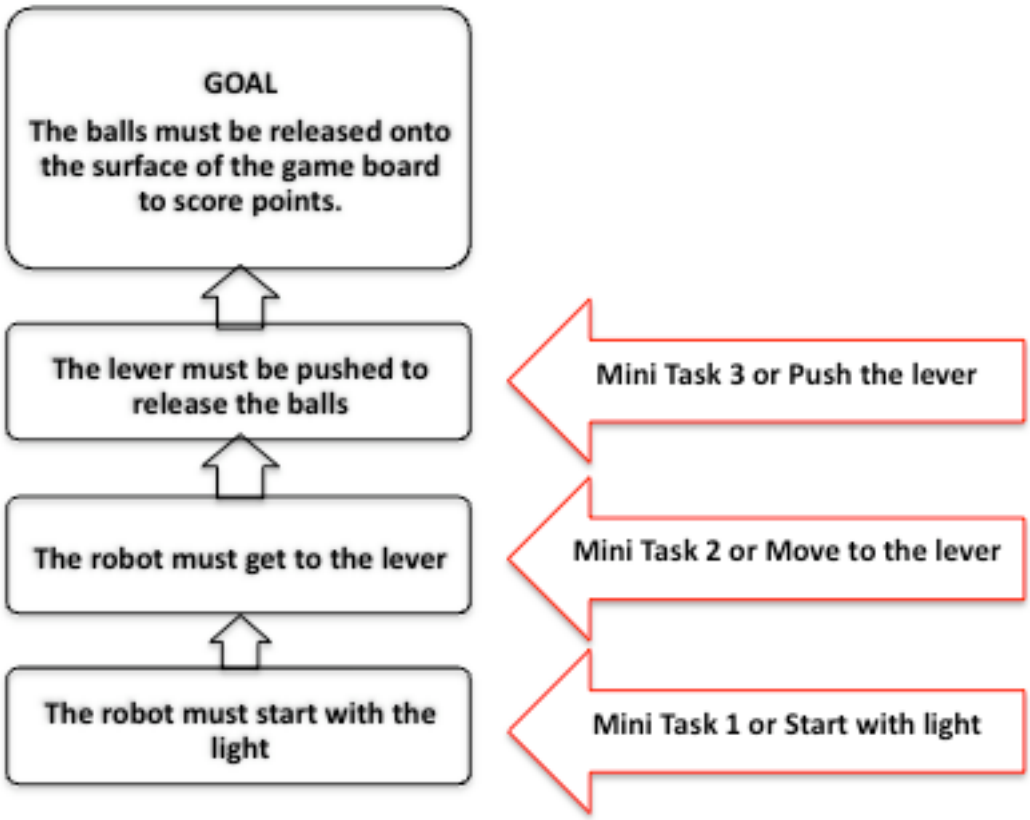
   This is a great whole team activity.

2. KIPR always includes multiple tasks that score points. **There is always a relatively easy way to score points. DO NOT overlook this**. Often times the game is set up so that by using simple **dead reckoning** (going a set distance or moving at a set velocity for a set amount of time) a team can score points. The ultimate goal of course, is to use multiple sensors to make intelligent robots, but using dead reckoning as a starting point is a good idea. For example; Robot A drives straight for X seconds or Robot B goes to this position.

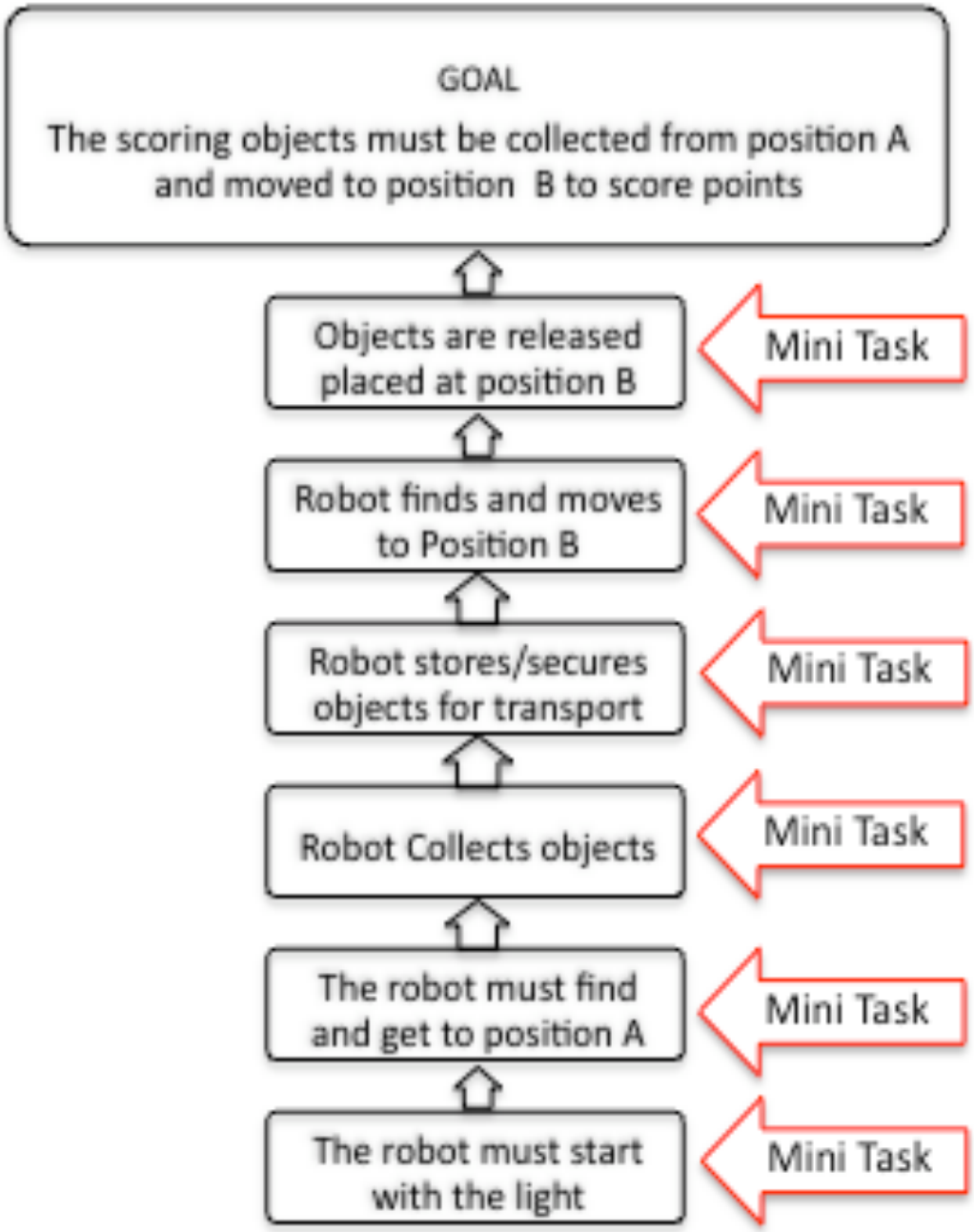
   Many teams are very successful because they get the simple, "easy points" consistently every time their robot runs. **Remember, scoring a few points consistently and having success is better than being unsuccessful and scoring no points.**

3. Start with the **"<u>easy points</u>"** by having the students discuss and document **<u>what has to happen</u>** to score these points (goal) by working backwards (Task Analysis) from the desired goal. This is why the goal is at the top and the start is at the bottom in the chart below. Working backwards helps the students focus on the goal and the step-by-step, sub task or "Mini Tasks" they have to accomplish to complete the final task.

For example: What has to happen - Simple Task A (easy points)



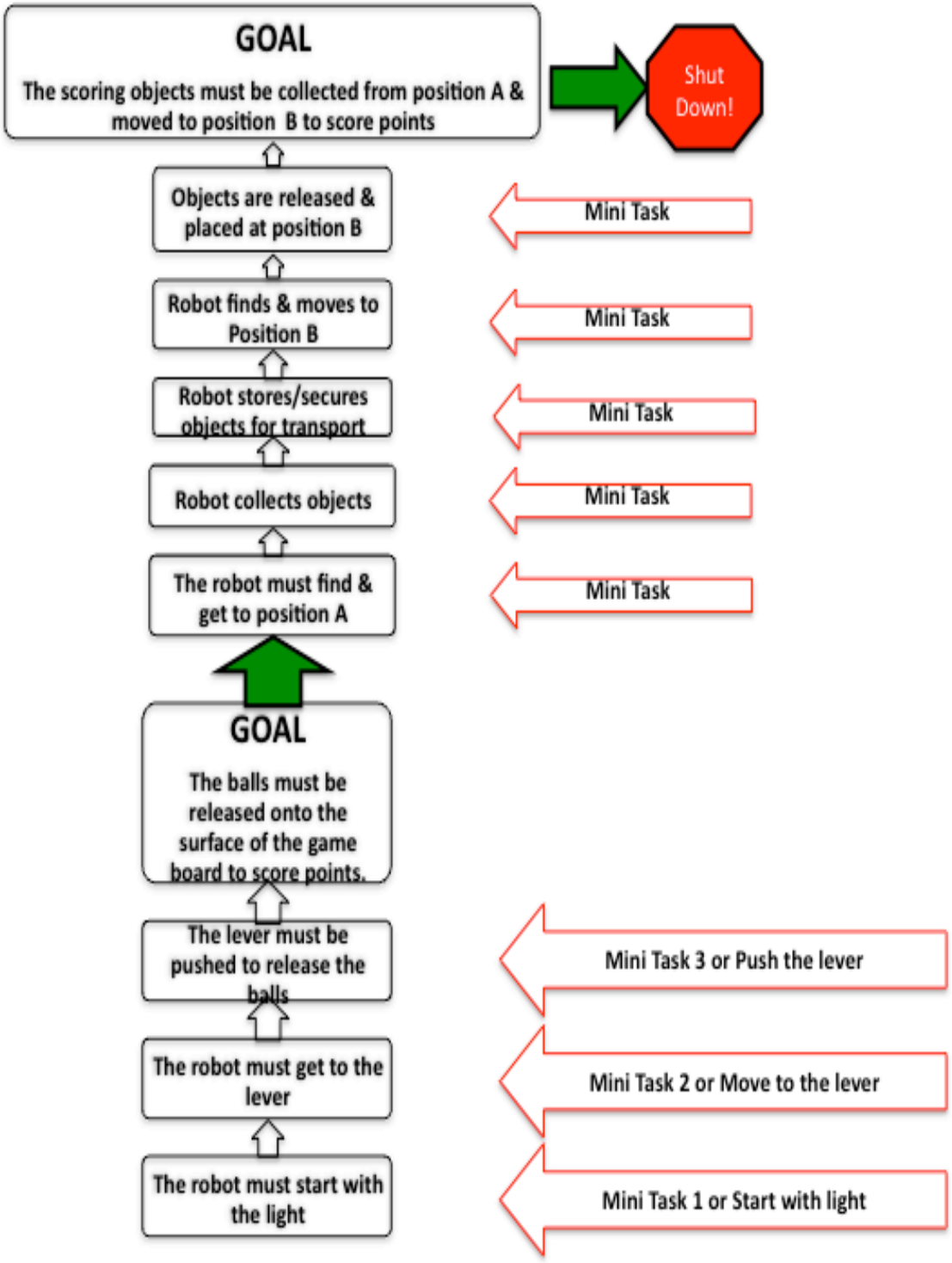
4. Then move on to the more complicated, “What has to happen” - Harder Task A (harder points).



It should become obvious that some of the tasks (goals) are more complicated and have more mini tasks or steps to complete.

5. When students want the robot to do task A and then task B and then task C the charts add together making the mini tasks needed to accomplish both summative and dependent upon one another. This is okay as long as they start with task A and prove that it can be

accomplished by reliably and consistently completing each mini task along the way before moving on to the first mini task of Goal B. **REMEMBER THE LAST TASK** is always to shut the robot down or it will be disqualified.



The students will have no basis to predict how long it will take them to complete mini tasks. In the end, if they are successful in completing only one task (goal) A, they have been successful and most likely will be competitive at the tournament. If they only complete some of the mini tasks leading up to the goal they can still have success (the robot started with the light or the robot found the lever, etc). The alternative may be a robot that partially completes or cannot complete any of the mini tasks or the goals leaving the students frustrated.

Setting the goals and mini tasks required provides the starting point for your students and an easy metric for evaluation of progress. In fact, the first software documentation submission is an exercise in task analysis (the code says this, resulting in robot doing this).

6. **The students have arrived and asked; what do we do?** Have you got your robot successfully and reliably completing mini task 1 or completing the wait for light and starting? You should have the students use a metric to determine success. (It must wait for the light to start and work 10 out of 10 times or 5 out of 5 etc.)

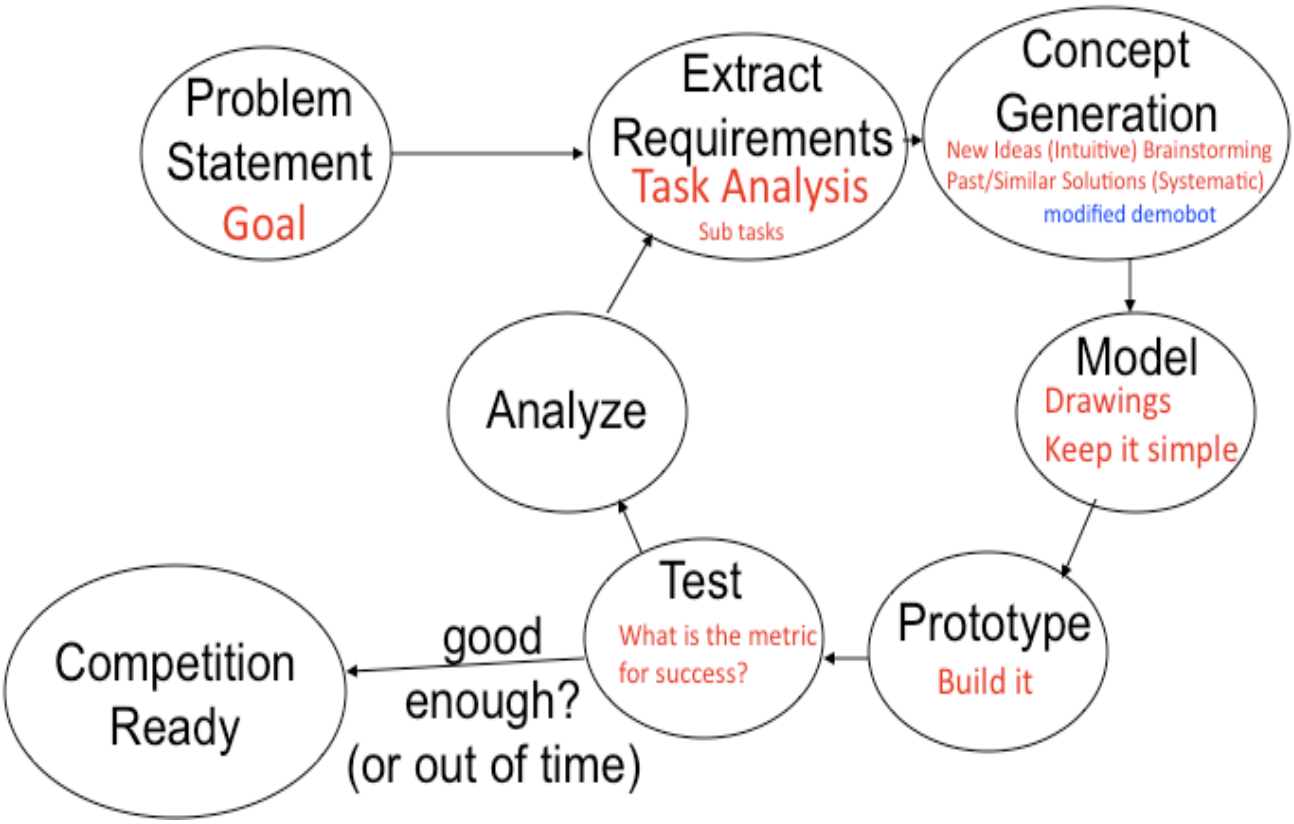
7. This is a good time to talk about precision and reliability. **Waiting for the light to start** and **shutting down at the end** of the round **MUST** work **100%** of the time or you face disqualification.

   Finding the lever on the other hand may work if your program gets the robot to within plus or minus 2 inches. In that case don't waste your time tweaking the program making sure it comes out plus or minus 1/16 inch. It could be that you can decrease the precision required in the programming with a mechanical solution. For example, building a larger structure to trip the lever could mean your robot can now be plus or minus 6 inches. Often times it may be beneficial to have students working on the programming to get the robot to the desired location while other students are working on the concept of a claw to perform the task. They can decide on the best concept and then move to prototyping by building and testing the claw **UNIT** to make sure it works before they mount or **INTEGRATE** it with the robot. Just because the concept and prototype works great by itself doesn't mean that it can be easily mounted on the robot. Once the integration is fixed and tested, they can then test and analyze the **SYSTEM.**

Many students will successfully complete the mini task once and think that is sufficient. Have them decide what the metric will be to accept reliable task completion before moving onto the next mini task. This is a good thing to document and possibly graph. This will come in handy when they are working on documentation submissions and onsite presentations as well.

If the task is continually giving them problems, reevaluate. Maybe a mechanical adjustment will make it work or if it is in the program, have they double checked their code, looked at the workshop examples, asked for help on the community site or called KIPR?

8. The engineering life cycle provides an overview of the entire process.



9. Students may have the robot "almost completing the task" and will want to move on to the next mini task, thinking that they can come back and fix it later. Often this leads to a robot that may complete a few mini tasks but completes 0 of the original goals meaning there are no points scored.

10. Another item to watch out for is "student intervention" as the robot completes the task but not autonomously. It works only if the student corrects it mid course or moves the game piece a little bit or…

    This is bad science and bad practice. The students are not allowed to touch the robots during the competition from after the hands-off period until the judges are finished scoring the round. If the students practice and are in the habit of touching and adjusting the robot, that

is how they will perform on tournament day. Practice like you are competing and compete like you practice.

11. **Use two robots** and plan so that if one doesn't work, it doesn't prevent the other one from working. If one disintegrates on the way to or at the tournament you can still score points. After your team has completed the task analysis for the current year's game, the students can pick an easy task for each robot. Remember, it is a good idea to go for the easier points reliably that often require less mini tasks or steps to complete.
Dividing the goals into subtasks will help with the overall management, because you and your students know what they should be working on and they have a good idea of how many mini tasks they have to complete before the goal is attained. If they are successful, you now will have a team that is completing two tasks with two robots and they may be very competitive at the tournament.

    The decision now is which robot is the best choice for which task and why? HINT: The Create platform is a heavy, fast platform and can spin out, slide on, stop etc. This needs to be taken into account when programming it. You may need to stop short of the objective to account for the slide/skid or maybe don't go full speed. Maybe the task calls for speed or maybe it calls for slower more precise movements.

    It looks like another brainstorming session is in order discussing the pros and cons of each type.

    What, you don't have the robots built yet? This is where the demobots come in handy.

Using the Task Analysis and demobots to program and build.

1. The catch 22 for students new to the program is that it is hard to program a robot that is not built (you can use the simulator to help) and if you take the majority of the time building a robot, there isn't a lot of time left to program it. Fortunately you have the demobots, which the students can begin to program right away **IF you don't take them apart right after the workshop.** You do have the build instructions in your workshop slides if they do need to be rebuilt. With the iRobot Create especially, structures (arms, claws etc) can be easily added. For a new team you should think about using the basic demobots to begin programming on the simple tasks and then simply modify the structure if needed to fit the task.

2. Using a demobot for Task A.

    If you are looking for activities/curriculum, you now have the means to create your own that is relevant because it ties to the current game. Mini task are individual activities that build on one another. If you want something to do off-season, during a camp or before the Global Conference, use a previous year's challenges or create your own. You can make the task more complicated as the student's abilities increase. Now, not only do you have to find the scoring objects and move them from A to B but you must sort the objects, placing the round ones in area B and putting the square ones in area C. The next task has the areas raised 10" above the level of area A etc, etc. You get the idea.

The examples of programs provided in the workshop can be used to help complete mini tasks. The wait for light code is provided, and the shut down after XXX seconds is also provided. Moving the robots forward, stopping, turning, backing up, moving in an arc are all provided. These can be used to complete a mini task such as get to the lever or find this area.

The flow chart from the task analysis can now be used as a road map to direct the programming and mechanical design.

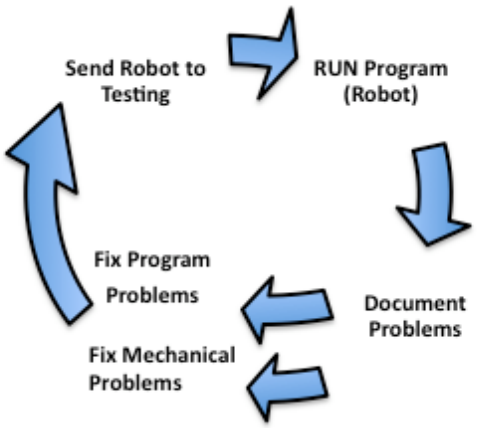
Where do we start? Well of course you should start with the first mini task of the goal you are working on. Again, remember you have two robots so the teams can be working on two separate goals. It is important that the students remember that both robots have to run on the same side of the board so they must choreograph their activities. This is when you need to bring the two groups together so they can work out any possible interference between the two. Often, teams will have one robot wait to start until the other one completes a few mini tasks or goals and moves out of the way.

You can begin by programming the existing demobot to wait for light and start mini task 1. The mechanical side of this is; ultimately, the light sensor needs to be mounted firmly in a stationary position so that it is easy to shine the starting light at it. This may change as modifications or additions are made to the demobot. I have seen numerous students at the competition who are holding the sensor up to the light or holding it in place while calibrating and then they let go, the sensor moves and they wonder why their robot doesn't start.

The students can quickly move to mini task 2 by moving the robot to the desired location with little or no adjustments to the demobot. They will discover soon enough that many factors come into play when they are moving the robot and it is not as easy as they imagined to drive to the same location every time.

Does it go straight or does it turn? Do you need to speed one wheel up and slow the other down to correct for this? Does the robot get hung up on something on the surface? If so, this needs a mechanical modification. Is it fast enough or too slow? Why doesn't the robot end up at the same place every time?

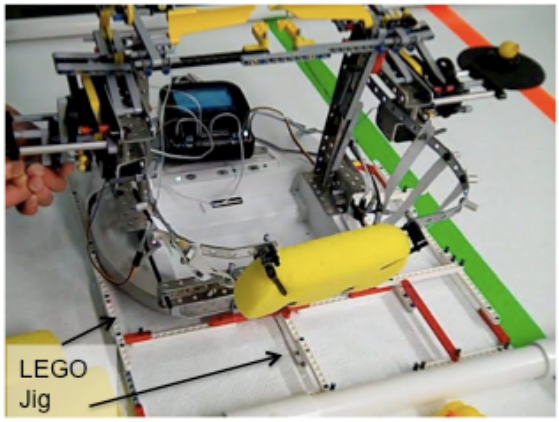
This is all great problem solving that the students will need to go through to figure it out. As a facilitator you don't tell them the problem is one wheel is turning faster than the other or that they are mounted at an angle, you help them reason it out. Why do you think it is turning? What could make it turn? The process works like this:

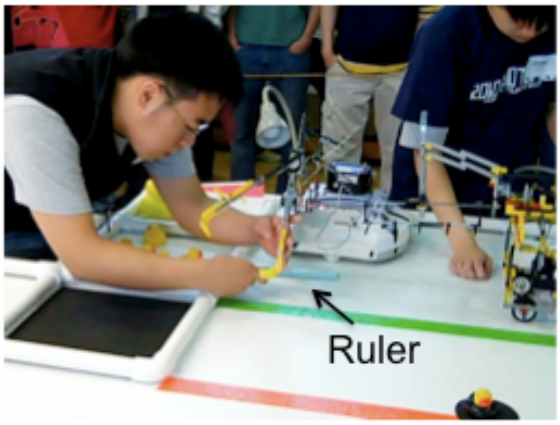


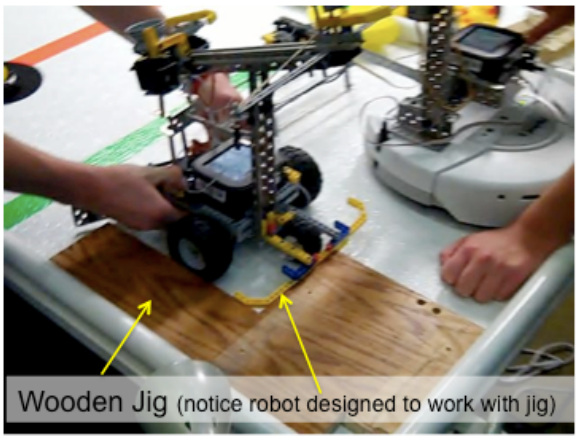
After some work they will hopefully get the mechanical problems ironed out and continue with the programming fixes until they have success.

Once they have it fixed and the robot running as desired it still may not consistently hit the mark. This is a perfect time to interject a little science and experimentation into the process. A common pitfall is ignoring the importance of controlling the starting position of their robots. This is especially true if they are using dead reckoning to navigate (as the students skill levels increase they can use more sensors and build smarter robots so that the starting position is not as critical). If you don't place the robot in the starting box exactly the same (controlled) every time, why would you expect to get the same results? This is especially true if the robot has to travel some distance or make several turns prior to reaching the area. The alignment error at the start is magnified at the other end.

There are many techniques to help align robots at the start of the round. Try measuring with rulers, utilizing jigs, or using the structure of the robot to align itself. In some Botball games, teams are allowed to place select game pieces on the board; in this case exact placement may call for measurement or a jig. Consistency is the key. You must also remember that you have a limited setup time in which you are required to set up your robot without incurring a penalty. A well made jig can speed up the process. There are some previous Global Conference on Educational Robotics papers discussing this. They can be found on our website.

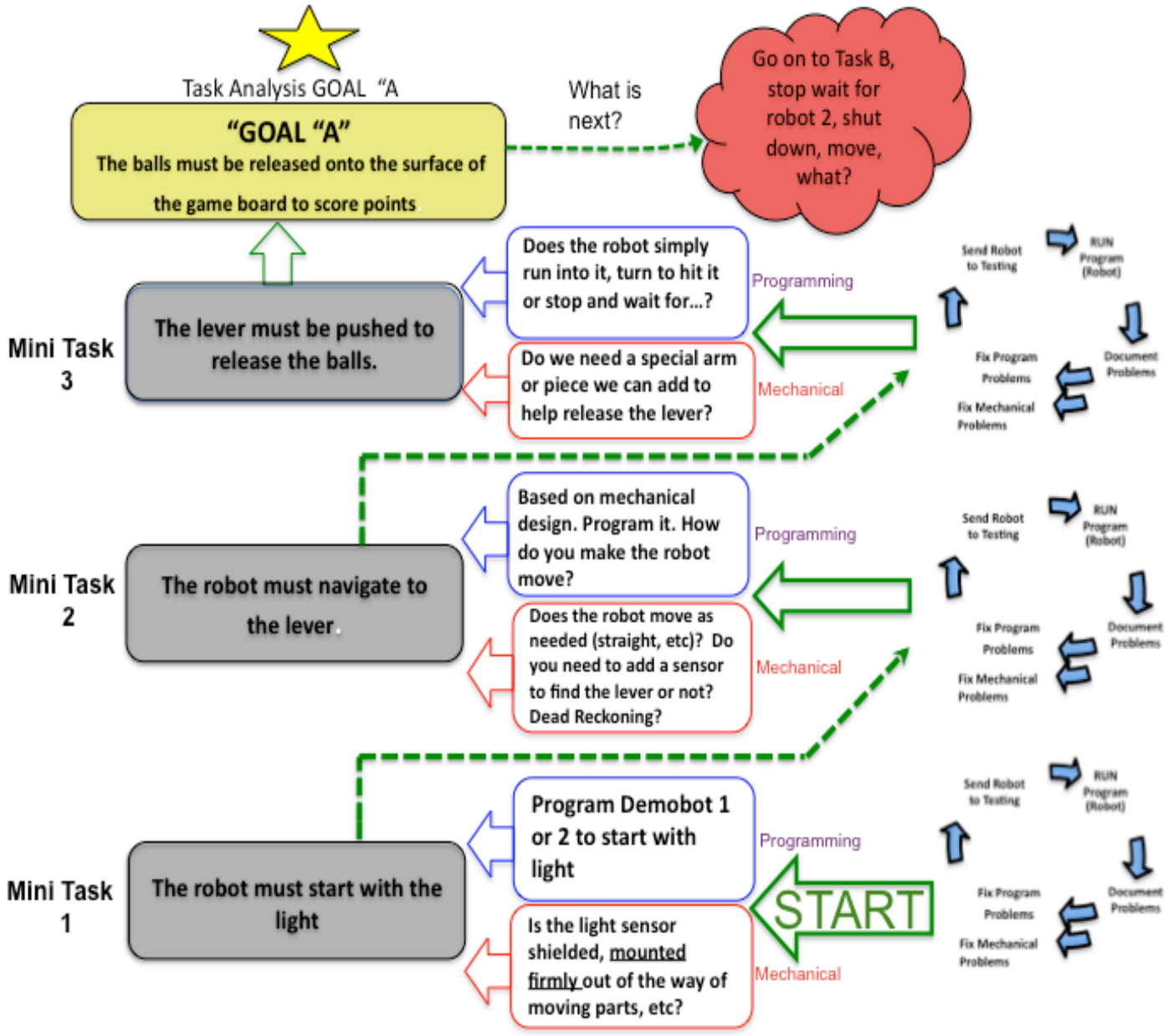






On to mini task 3. This is where some mechanical adjustments to the robot may need to occur. Remember **simple designs** are easy to build, integrate into the existing structure and ultimately program. Once this works and the robot starts with the light, moves to the position and releases the lever, Task A is complete.

3. The following is a graphical representation of how the process can work. They are not isolated but dependent upon one another. Someone can be working on the program while someone else works on mounting the light sensor etc.



Using Trial and Error

1. Students love to use simple trial and error. Well that didn't work, let's change this and try it again. This is a good way for students to begin to understand how the programs they are writing change the robot's behavior. The downside to doing everything by simple trial and error is that it is extremely time consuming and there is a better way.

2. **Use MATH**. If the student wants the robot to go forward 100cm, they can play with the code (trial and error) until they are successful. However, they can also be encouraged to understand that the distance traveled is a function of how many times the wheel goes around and the circumference of the wheel being used. Now they are beginning to get an understanding of the concept.

3. **Record DATA**. Have them read and record sensor values to get a feel for them prior to programming. For example, if they are looking for a black line with a reflectance (top hat) sensor, what value do they get from the white background and what value do they get from the black background? They might want to record this for future reference. (Note: values change due to lighting conditions; it is guaranteed that the tournament lighting will not be the same as your classroom.) Turn off some of the lights and take the values again, does this affect the values? Does the distance between the surface and the sensor make a difference in the value returned? Sounds like a great experiment and activity. Have the two surfaces, control the distance form the surface (say ¼") and then read the values. Measure the values at increasing increments away from the surface, say up to 2" in ¼" increments. Now you have some information that can be graphed and may be very useful when mounting this type of sensor on a robot. Sound like good information for a documentation notebook.

4. **Control Variables**. Students working on the robot are trying to figure out why it is not behaving as predicted. Using trial and error they run it, then change 32 different things in the program and the mechanics and it doesn't work, repeat, repeat, and repeat. Or maybe it does work, but they have no idea what actually made it work so if the same or similar problem occurs again they are clueless. Isolating possible causes and then testing them one at a time is the answer. Just change the one item and see if there is an effect. At least you will have an idea of what fixed the problem or made it work for future reference. Control the variables except for the independent one you want to test.

Now you have the basics to get your team started. If you haven't finished reading the "Hints for New Teams" document you should do that now. That document will give you the hints and tips you need to prepare your documentation and what to expect at the tournament!

Good luck!