How to Get a Perfect Score
On Online Documentation
Grant Uebele, Alex Stahl
Dewitt Perry Middle School
Bg.uebele@verizon.net, sike846@gmail.com

How to Get a Perfect Score On Online Documentation

1. Introduction

Online documentation is one of the most under-estimated parts in the Botball competition, and one of the most important. Documentation - the thing you leave until the last minute. Documentation is more important than some people think. Documentation counts for a third of your overall score. That means that if you get a bad score on your documentation, you will probably get a bad score overall. Together, if you get a perfect score on documentation, a really good score on seeding rounds, and a good score on double elimination, you will probably get in the top five.

Below is the algorithm for the overall score. Notice how the documentation really contributes.

Here, you can see how the documentation score contributes

$$\text{Doc Score} = 0.2\left(\frac{n - P_1\ \text{Rank} + 1}{n}\right) + 0.23\left(\frac{n - P_2\ \text{Rank} + 1}{n}\right) + 0.27\left(\frac{n - P_3\ \text{Rank} + 1}{n}\right) + 0.3\left(\frac{n - \text{Onsite Rank} + 1}{n}\right)$$

$$\text{Seed Score} = \frac{1}{2}\left(\frac{n - \text{Seed Rank} + 1}{n}\right) + \frac{1}{2}\left(\frac{\text{Team Average Seed Score}}{\text{Max Tournament Seed Sco}}\right)$$

$$\text{DE Score} = \left(\frac{n - \text{DE Rank} + 1}{n}\right)$$

n = Number of Teams at Tournament

Documentation Score
\+ Seeding Score
Double Elimination Score
= Overall Score

*Overall score will be between 0 and 3

Here, we will demonstrate the difference between an online documentation score of 85 and a score of 100 by placing them and their corresponding rankings into the equations above. We will assume constants in the onsite rank (3rd), number of teams (34), team average seed score (200), seed rank (3rd), maximum seed score (280), and double elimination rank (5th).

1.1 An Overall Score with a Documentation Score of 85

Assuming that an online score of 85% for each of the three periods gets you 15th in the first period, 19th in the second period, and 20th in the third period, you can calculate your total score:

P1 rank= 15th P2 rank= 19th P3 rank= 20th Onsite rank= 3rd *n(number of teams)*= 34
Team average score= 200 Seed rank= 3rd Maximum score= 280 DE rank= 5th

So, if you got an 85 on online documentation:

Doc score: 0.2((34-15+1)/ 34) + .23((34-19+1)/34) + 0.27((34-20+1)/ 34) + 0.3((34-3+1) /34) ≈ .6274
Seed score: ½ ((34-3+1)/34)+ ½ (200/280) ≈
.8277
DE score: ((34-5+1)/ 34) ≈ .8824

Overall score: .9188 + .8277 + .8824 ≈ 2.33744

1.2 An Overall Score with a Documentation Score of 100

Assuming that an online score of 100% for each of the three periods gets you 1st in the first period, 1st in the second period, and 3rd in the third period, you can calculate your total score:

P1 rank= 1st P2 rank= 1st P3 rank= 3rd Onsite rank= 3rd *n(number of teams)*= 34
Team average score= 200 Seed rank= 3rd Maximum score= 280 DE rank= 5th

So, if you got a 100 on your online documentation:

Doc score: 0.2((34-1+1)/34) + 0.23((34-1+1)/34) + 0.27((34-1+1)/34) + 0.3((34-3+1)/34) ≈ .982
Seed score: ½ ((34-3+1)/34)+ ½ (200/280) ≈ .8277 DE score: ((34-5+1)/ 34) ≈ .8824

Overall score: .982 + .8277 + .8824 ≈ 2.69244

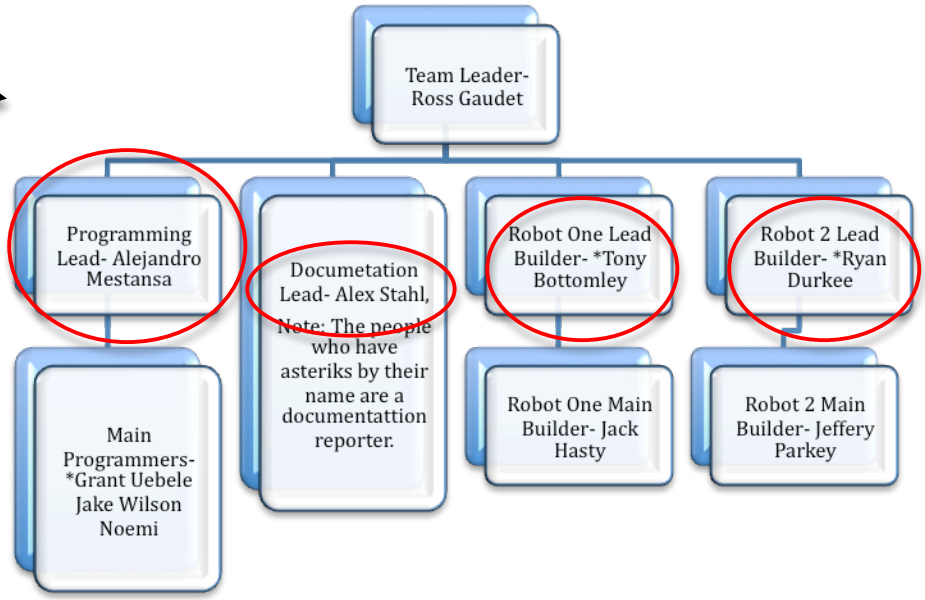
1.3 The Difference Between the Two Scores

Although it doesn't look like much of a difference, the slightest tenths can change your overall rank. Many teams have won because of the slightest tenth. For example, at the regional tournament in Oklahoma, the winning team had a score only two tenths higher than the 2nd place team.

2. Organization

Organization is very important when it comes to getting a perfect score on documentation. It helps strengthen the team and the work the team does. In organization, hierarchy has a big role. It helps you decide who is doing what. You should have a team that builds the robots, a team that programs the robots, and a team that does documentation. Also, a thing that our team did was we have a person on each team that reported to the documentation team and told them the progress of their work. Basically, you are splitting the team members into sub-teams, depending on what they were good at.

As you can see here, we have split up our team into three sub-teams as explained above. We have a team leader for every sub-team. This helps get tasks done faster.



3. Documentation Itself

Now, what you've all been waiting for - the documentation itself. The most important aspect of documentation is being able to follow the rubric. The rubric tells you exactly what you need to do to get a perfect score, and it really isn't that hard.

There are some things that are common to every submission, like, for example, all files must be submitted in a .PDF format (except for commented code). An updated project plan should be part of each submission. All of the files have a strict file name that must be followed and the submission must be a single file.

3.1 First Submission

3.1.1 Project plan

To start off your project plan, you have to decide a strategy for your plan. Our team used Excel. Before-hand, you should have already made your hierarchy. Know your due dates and use that to set your goals. When you set your goals, consult your rubric to set your sub tasks. Color-coding helped our team out a lot. The whole idea of a project plan is to help your team advance. When you color code, your team can see what has started, what is finished, and what needs to be completed when. Make your plan easy to read. Include weekly comments to explain what you have done.

3.1.2 Design Concepts

You must include multiple concepts of your design. For each concept, put the maximum number of points a concept could score. Put your rejected concepts and an evaluation of your robot. Include lots of details and key features.

3.2 Second Submission

In this section, include how and why you have designed and built your robot to do something, instead of merely saying that it carries out a task.

3.2.1 Mechanical Design

Include lots of pictures - pictures of prototypes and every time your main robots experience a change. Include tests of these parts and descriptions of each part.

3.2.2 Software design

Include all of the goals you want this robot to accomplish. Say the functions needed and how to perform that goal. Include the testing of that goal and how many tests you will need.
Have a flowchart of how you will complete a certain goal or task.

3.2.3 Project plan 2

If you laid out your project plan right, this should be a piece of cake. Finish up your goals for week 7 if you haven't already.

3.3 Third Submission

3.3.1 Design Changes

Of the three documents that consist of the third submission, Design Changes is probably the easiest document of them all. Here, you just list any changes you made to the robot, and insert pictures of them. Make sure your document has a detailed description of the change to

accompany the picture - why you had to make the change and how it works after the change. Also, if you get points for more than one change, like this year they asked for you to document 3 changes, be sure that you do all three. If your robots were anything like ours, we made probably a hundred changes over the 7-week period. At the very end of the document, be sure to add an elegant solution to a problem you have been having due to your design.

As you can see, our team has a detailed caption to accompany every picture.

First change: Wheel Mount

We had to change the wheel mount because it kept falling off. Originally we had screwed in two large black gears into a servo horn (you can see one of them in the picture). This gave the axle lots to slide into. Our wheels bowed out badly because of this, the robot sagged and this caused it to not move very well and end up in different places. We ended up using only one black gear screwed into the servo horn. Although the axle does not have as much to slide into and stay secure the wheels move closer to the n=motr and don't bow out.

Here we have an elegant solution to a problem that we have been having with our designs

Elegant Solution: CBC

We are most proud of our ramp and ramp dropping mechanism. We needed a way of getting the oily ducks into the oily duck center. We built a ramp (over and over again), but needed a way to put it down. We had lots of ideas, we finally just set it on the back of the create with only a motor with a piece of lego that clamps it down. When the create drives up to the place it drops the ramp, all that happens is we reverse that motor and the ramp falls right off. Easy! It is really cool and amazing to watch!1

3.4 Commented Code

Ah, commented code, in my opinion, the hardest part of online documentation. This, for example, is something that you don't want to leave to the last minute. I did that very thing: staying up until eleven o' clock the night before it was due. Not only did I have to write many, many of the comments, but I had to actually figure out what the robot was supposed to do first. Here's a tip, don't do what I did. Comment your code as you write it. This will help eliminate the difficulty and time consumption necessary to produce this document. The comments will actually serve the purpose that they are meant for – telling the original programmer and all programmers looking at the code later, what the robot is being told to do. Commented code is a document that if you stick to the rubric, you'll be fine.

There are two main parts of commented code that you need to keep in mind: the headings and the line comments. The thing is, you have to comment your code and tell where the robot is, and what it is doing. A suggestion is that you should divide your code into large groups. For each group, you should have a heading that describes what task that certain group wishes to perform. This will help organize your program, and will get you some of those valuable points.

In your code, at the very top, you should list the authors, what the program does, and all of the dates when you changed or added to the code, or accomplished a task. Another good thing to do

is to group the servo definitions and function definitions according to what they do. You also need to make sure that each function definition is preceded by a detailed description of what the function does. Make sure function and variable names are descriptive and convey their use in the program. Make sure your comments use standard spelling and grammar.

```
//Servo definitions for arm used to scoop and push clean ducks
#define ARM_PORT 0
#define ARM_UP 1600
#define ARM_DOWN 25

// Servo defintions for scoop to gather and manipulate the dirty ducks
#define SCOOP_PORT 3
#define SCOOP_UP 1800
#define SCOOP_DOWN 150
```

Remember, it will really help you if you comment every line of code as you write it.

```
my_move(-40,-200);// moves away from the wall opposite the duck pond
my_turn(-85,150);// turns to face the dirty ducks and oil slicks
scoop_down();// puts the scoop down
my_move(300,400);// moves forward and collects first pile of ducks if there
scoop_up();// raises the scoop
my_turn(-81,150);//turns to face dirty duck center
wall_bump(5.0,300);// bumps into wall of dirty duck center
my_turn(78,150);// turns to face clean ducks in wet lands
my_move(-150,-200);// moves towards clean ducks
```

Lessons learned:

It's the end of the online documentation period, time to get the bottle of soda! No, unfortunately, the period is not quite over. It's time to share some things you've learned during this stressful time. In fact, ask the whole team about what they learned. Our team took a survey of all the team members. We then put all the lessons together, and submitted it. Don't forget to mention what your team gained from the documentation process and any surprises. Ah yes, you should include a section that focuses on advice for future team members. So, just ask your teammates, write it down, and sprinkle in some advice and POW! It's that simple. So remember:

Include lots of lessons that you have learned during the year:

Experience Gained

This year, our experiences were valuable. One member of our team reported that when programming or testing don't sit around and do nothing; fix the problems and try again. Robotics teaches you a lot of things about programming, like different functions and techniques. We learned C programming and how to relate it to a robot. Programming is quite difficult, but it is rewarding. Botball also teaches how to build parts of robots, like claws, scoops, and mounts and make sure they are strong and will not fall apart. When changing ideas, don't waste time. Botball also teaches young students to work cooperatively with their teammates, this was one of the hardest things we learned.

All the things gained:

Documentation Process

When asked about documentation, not many people seemed unenthusiastic, but it has to be done, and according to some people <u>IT NEEDS TO BE DONE EARLY!!!</u> You shouldn't wait till the last minute to do documentation. This can be the easiest thing in the world, or the hardest. Don't underestimate documentation; it is a lot harder than it sounds. It can also be very stressful at times. Use the rubrics and examples, they show you exactly what to do. For the documentation presentation, pick the people who are most knowledgeable about the robots.

Advice for the new kids!

Advice for Future Teams

If someone from your team isn't doing something productive, don't get mad at them; encourage them to do something.

A lot of team members said that you have to learn how to catch a robot.

When working, you need to concentrate and work hard.

Your mentors are smart and many time right, so listen to them.

Try not to get impatient with your teammates.

Remember to "Keep It Simple Stupid" (KISS).

4. Time Management

Documentation: a crash waiting to happen, a bomb waiting to go off. These all could describe documentation. The fuse to it all: leaving it to the last minute. Let's just say it will be the worst thing you will experience during the whole robotics experience. There is a way to avoid this catastrophe; it's called… time management. Make sure to set dates for when you want to work on the documentation. Make sure you plan early. Our team assigned people to certain parts of the submission. This was a whole divide and conquer thing. As you can see, it helped us. Also, make sure you always, always, follow your project plan. Time management is very valuable.

5. Working Together

5.1 Team Members

The best way to have a successful robotics team is to have good teamwork. With a cohesive team you can get a lot accomplished in a short amount of time. If you become friends and don't fight your team will succeed. Fighting could injure someone and get you kicked off the team. It also wastes precious time. Another good idea is to identify the strengths of each team member and assign tasks that utilize those strengths. Everyone will be generally happier.

5.2 Sharing Ideas

Ideas are a very important part of your success as a team. You, as a team member, must pitch in your ideas so that your team is successful. Go around in a circle asking everyone their ideas and write them down. From the ones you picked out, list the pros and cons of each and choose the best one. Review your ideas with your mentor.

6. Conclusion

At the regional competition in Oklahoma this year, our team scored third overall. The reason this score was so high was probably the fact that we got a perfect online documentation score. This shows how a documentation score can really affect the overall score. Hopefully we have given you information in this paper that will help you get a perfect score in future documentation periods.