Fake Book Addendum 1:
Game Simulation and Estimation

In FRC we often have to set important directions without having all the information needed to do so. Early in the season, having the best guesses possible of how the game will play is incredibly important. You need to start somewhere.

Game Simulations:

Teams often do these on Kickoff. I've heard of 3 forms of them:

1. Kids Run Around Acting as Robots (KRAAAR)

Find a space, and set up a play "field" in that space. Using placeholders as game pieces, act out a couple of matches (commonly on office chairs or floor scooters) while keeping score. Play out different robot archetypes. Try different alliance strategies.

The game played by humans is not at all 1:1 with what it will be as a robot. And it can be hard to keep track of and analyze what is going on. But it gets you in the ballpark. In particular, this method is effective at giving everyone a sense of space and how the field is laid out.

2. Autodesk Synthesis

This is a robot simulator. Plug in a controller and start seeing what it feels like to drive around that new field! Time yourself driving between the points of it, time yourself going through the motions of scoring game pieces, average the times, don't fudge the times, get estimates on everything. Open it up and play.

3. Bring Out Old Robots

Get on a carpet, mark up the field with tape. Take an old robot to the field, and see if it can play

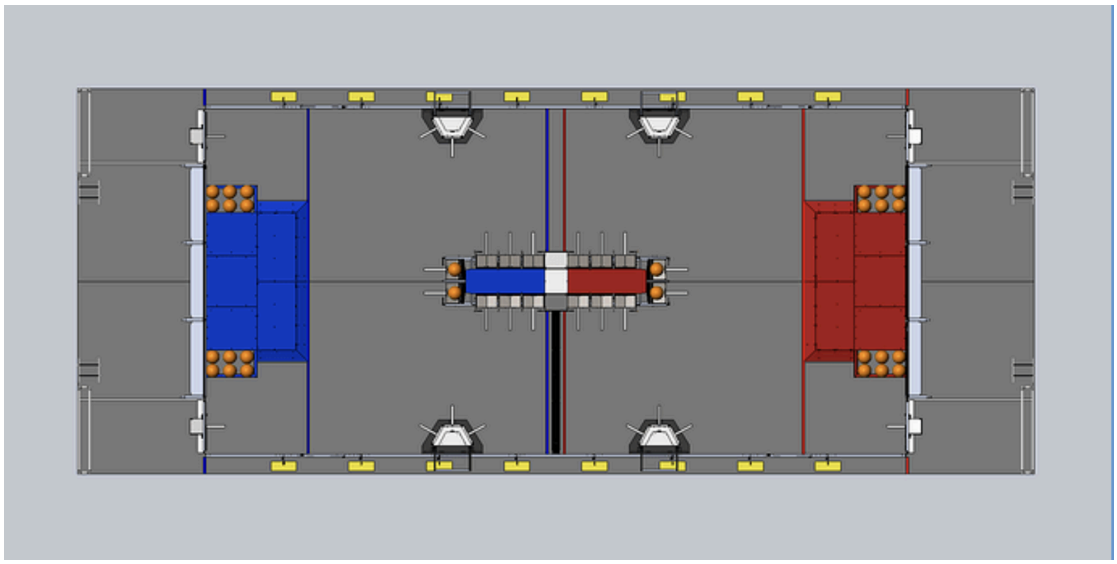
You can learn a lot from driving a base through the motions. Take a second robot, and try out some defense to see where the cycle can be stopped. You can start doing defense and crowding analysis right after the game is released. Once you see real robots driving around something that's *to scale*, you start to get a personal feel for the game.

ESTIMATION:

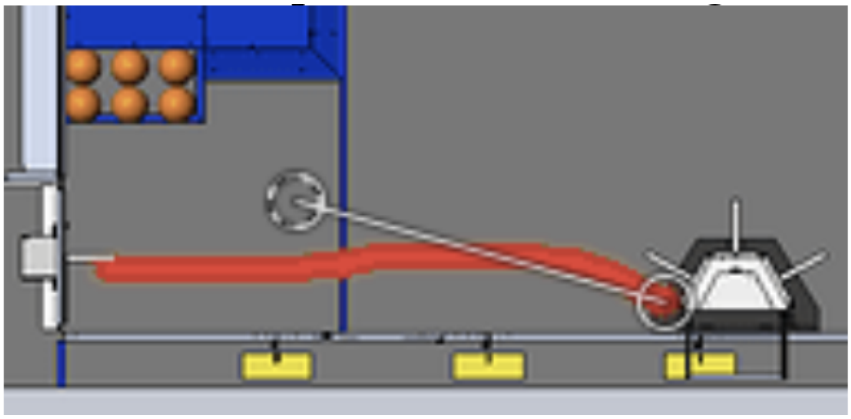
With estimation you are not acting out the game. You are doing napkin math to predict cycle times, match scores, and the relative value of tasks. Sometimes, you can strike surprisingly close to the real values of competitions.

2019 CYCLE TIMES

Take a look at the 2019 field



To begin calculating cycle time, let's draw a commonly driven path onto the field:



We will approximate this distance from the loading station to the near side of the rocket as 15 feet (You could get an exact dimension from the field drawings, or in CAD)

Here is what the events of near-rocket scoring cycle are:

1. Starting from the loading station, the robot drives to the rocket
2. An elevator is lifted to get to the desired level of the rocket and deposits the hatch
3. The robot turns back to face the loading station
4. The robot drives to the loading station
5. The robot intakes a hatch panel
6. The robot turns back to face the rocket.

These actions don't necessarily happen sequentially as I laid them out (especially now in the reign of swerve) But we will assume that they are for the sake of our math—this is all just a spherical cow. If you add up the time it takes to do all of those things and you will have the cycle time.

We can do this, relying on some rough assumptions.

Let's say the robot travels at 10ft/second. That means to travel to the rocket (step 1), under our assumptions it takes ~1.5 seconds (A sprint distance calculator is a better tool made for this)

How long does it take to lift an elevator and to place a hatch (steps 2)? Let's say based on past experience, the team thinks 2 seconds is a reasonable target. You could also watch previous game footage, or use a time-to-goal mechanism calculator (like re.calc, or JVN calc) for a base assumption.

Then the robot needs to turn back towards the loading station (step 4). I'm just making up approximate numbers again: .5 second.

Step 5, it drives back. 1.5 seconds. (Same as before).

We intake a hatch (step 6). We'll say this takes the same amount of time as scoring the hatch—2 seconds.

And then we turn again (.5 seconds) to face the rocket.

With that math, our predicted cycle time would be:

(1) 1.5
(2) 2
(3) .5
(4) 1.5
(5) 2
+ (6) .5

= 8 Seconds

8 seconds is our theoretical max near-rocket cycle time not accounting for driver error, defense, dropped game pieces, etc (This roughly aligns with real world Destination Deep Space—some teams could AVERAGE 11 second cycles during a match, and some of them likely got close to their minimum cycle time being 8 seconds if not lower…)

I did the same rocket calculation at the beginning of the 2019 season and came up with a cycle time of 15 seconds. (I thought the turning, intaking, and scoring would all take longer)—My conclusion was that soloing a rocket was a near impossible task only to be accomplished by the highest echelon of teams, and wasn't worth going for. There was some truth in the rocket RP being difficult. But I was a bit off.

Nonetheless, having some idea of the relative difficulty/time-span of tasks is better than none at all—you can always update your time estimates based on your prototypes and real driving.

PTS/SEC COMPARISON

Now we have an estimate at how long it takes to score a hatch, and we know how many points a hatch is worth, so we can figure out how many points we will get each second by scoring hatches: 2pts/8seconds = .**25 pts/sec**

If we assume that a cargo ball cycle to the rocket takes the same amount of time we would estimate: 3pts/8seconds = **.375 pts/sec**

(In reality, cargo balls often rolled closer and picked up faster)

One last thing: We want to know if it's even worthwhile to make a Hab3 climber—Why would a team build a climber if they knew they could score more points with their time doing either hatches or cargo? How fast does a robot have to climb for it to be worthwhile?

In Destination Deep Space, climbing is worth 12 pts. In order to be sure that it is worthwhile in any situation, we need climbing to be able to get us more than .375 points per seconds (which we can get from cargo). Some quick math tells us how quickly we must climb:

12pts/x seconds = .375 pts/second
X = 32 seconds

32 seconds is a pretty long amount of time! For reference, if you could climb in 10 seconds that's 1.2 pts/second. Hab3 was worth quite a lot in 2019.

Further Reading:

Compass Alliance Game Strategy Pathway
4481 Team Rembrandts 2025 Build Thread (Data-Driven Decision Making)