

Electric Catapult Design and Optimization

(2016 FRC Season, FIRST Stronghold)

Steven Spoldi, Mentor, FRC Team 230 Gaelhawks

Overview:

One thing we try to do at Team 230 whenever faced with designing a mechanism that needs to dispense a significant amount of energy is to try to perform a simple first principles estimation of how much energy or power we need, to determine whether or not our initial concept mechanism is up to the task. Walking around in 2016 (and even more so in 2014, Aerial Assist) I noticed a lot of creative concepts for launching game pieces, but what appeared to be less than optimal energy delivery. My goal here is to walk through the initial sizing and basic optimization we performed on our 2016 catapult in such a manner that teams can use this example as a template to perform their own initial sizing and design for their own mechanisms.

Developing a fully electric catapult has a lot of advantages: one mechanism for shooting and reset, low parts count (motor, gearbox, catapult arm, and angle sensor), and light weight. What's a bit trickier vs. a spring driven design is the selection of how big a motor to select, and what is the desired gear ratio from the motor to the arm to get the best results. Answering the last two questions will be my primary focus here. The goal is not to produce a finely tuned design, but to quickly get a design in the ballpark to determine if it can meet the performance criteria, and then to have the basic mechanism defined prior to committing a lot of time to prototype construction and test.

Basic Concept, First Principles Physics:

The basic catapult is pretty simple: an arm of length L pivoting around a point driven by a spring or motor to throw a projectile. The first step is to make a boatload of assumptions, some of which may or may not be all that good, in an effort to reduce the problem to a manageable level. In this case these are the basic assumptions we will use for our catapult:

1. The catapult will start in a horizontal position.
2. The arm will rotate 45 degrees at which the ball will leave the arm, traveling up at a 45 degree angle.
3. We will use the speed of the catapult arm as it swings through the 45 degree point to determine how high and how far the ball will go.

In reviewing these assumptions, #1 makes logical sense given how most robots would be laid out, and #3 given #2 makes sense as well, but why #2 at 45 degrees? Well, basic ballistics says that when launching a projectile at a given velocity the best angle for maximum distance is 45 degrees, it seems like a pretty good place to start. In addition, at the peak of it's trajectory a ball launched at 45 degrees will have traveled twice as far as it's maximum height and eventually travel four times that height until it hits the floor. This gives us a perfectly usable baseline for trajectory, we can always change our mind later.

How hard do we need to launch the game piece? This is the big question. The tower goals are roughly 7 feet high, and with our 45 degree trajectory gives us a shooting position of roughly twice that at 14 feet away from the goal. These dimensions play very favorably with the Stronghold field layout, so far an acceptable assumption. Going back to basic physics, we use the following:

1. $\text{height} = 1/2 * \text{velocity}^2 / \text{acceleration}$
and, with a little rearranging:
2. $\text{velocity} = \sqrt{2 * \text{height} * \text{acceleration}}$

where: velocity = initial vertical velocity ($0.707 * \text{launch velocity}$)

acceleration = acceleration of gravity ($32.2 \text{ feet/second}^2$)

(see the Wikipedia page "[Range of a Projectile](#)" for the relevant physics)

Working this out, we get an initial velocity = $\sqrt{2*7*32.2} = \sqrt{450.8} = 21.2$ feet/sec. Since this vertical velocity is equal to $0.707*$ launch velocity, we get the following:

$$\text{launch velocity} = \text{vertical velocity}/0.707 = 21.2/0.707 = 30 \text{ feet/sec.}$$

This is the basis for our design. At a minimum our catapult needs to generate 30 feet/sec to get the ball to the goal with our 45 degree launch angle. Since it's driven by an electric motor, it's easy to dial it back, but impossible to exceed 100%. So, first question, how much extra oomph do we design into our catapult to insure we'll have enough to get to the goal? My rule of thumb is to determine how much energy you need, double it, and start from there. Then, if your mechanism is 50% efficient or better, you're good to go. Remember kinetic energy = $\frac{1}{2} M*V^2$? To double the projectile energy I need 1.4 times the velocity, so our new target velocity = $1.4*30 = 42$ feet/sec. At this point it pays to be conservative, coming up short in week 5 of build season is not where you want to be, so in this case more is more.

Catapult Design & Simulation:

This is where it starts to get a bit interesting. Here is the prototype catapult arm we started with for the 2016 robot:



Prototype catapult arm, FIRST Stronghold 2016

The arm is about 28 inches long, the longest arm they would let us fit on the chassis (at the time we made it, we did not have the sprocket, as we didn't know the gear ratio). Armed with that info, let's take a short side track and look at a spring powered design.

Since a spring powered catapult is probably the most popular, let's figure out how much spring or bungee we need. If the ball sits 28 inches away from the pivot (2.33 feet), then we get the following distance traveled when the catapult moves through 45 degrees:

$$x = \text{radius} * \text{angle}, = 2.33 \text{ feet} * \pi/4 \text{ radians} = 1.83 \text{ feet}$$

Note that 45 degrees = $\pi/4$ radians. This is where using natural units like radians and radians/sec start to pay dividends. Everyone likes to think in degrees, but when you go back and forth between angular and linear systems, or start to do some calculus, radians start to make your life easier.

At this point we need to get a few more parameters. The boulder weighs about 0.65 pounds, converted to units of mass (slugs) = weight/gravity = $0.65/32.2 = 0.020$ slugs. I cannot stress enough how important it is to use appropriate units. Everyone at the shop makes fun of slugs, but that's the only way equations like $F = MA$ work. Going back to our 42 foot/sec velocity target, remember the formula for kinetic energy:

$$\text{energy} = 1/2 * \text{mass} * \text{velocity}^2$$

So, energy = $1/2 * 0.02 * 42^2 = 17.7$ foot*pounds of energy. If we have a constant force spring for our arm, remember that energy = force*distance, or $17.7 = \text{force} * 1.83$ feet (remember the catapult travel distance?). Rearranging we get force = $17.7/1.83 = 9.7$ pounds.

After the dust settles, we see that we need about a 10 pound spring on a 2.3 foot arm to launch the 2016 boulder 7 feet up and 14 feet away. Very handy information to have as I stroll to the back room to get some bungee material.

Electric System Design:

Not having a quick rule of thumb for sizing the motor or gear ratio for an electric version of the catapult, we are forced to determine the results via simulation. Fortunately, we can put together a simple iterative simulation in short order. We will develop the basic equations that can be used with either a general purpose programming language or spreadsheet to calculate the numbers we need, but first we need to understand the basics of how a DC motor works (virtually every motor legal for use in FRC is a DC motor).

Everyone is familiar with the equation $F = MA$. Since we have a rotating system, we will use the rotating equivalent $T = J\alpha$, where T = torque, J = inertia, and α = angular acceleration. First we will determine motor torque, then acceleration, then velocity, and then position (all angular). There are 2 very important parameters associated with any DC motor, they are stall torque, and noload speed. In a nutshell, when you apply power to a motor that is not rotating, the initial torque the motor generates is the stall torque. As the motor speeds up, the torque goes down until it reaches 0, which occurs interestingly enough when the motor speed gets to it's noload speed. The full equation looks like this:

$$\text{torque} = \text{stall torque} * (\text{noload speed} - \text{motor speed}) / \text{noload speed}$$

Since we will drive the catapult arm through some gear reduction (the value of which we will determine later), we need to decide where we will simulate, at the motor (motor acceleration, speed, and position), or the catapult arm (same variables there). Either works, but I'm going to assume it's easier to visualize at the catapult arm, so that's what we'll do. At the arm, we need to know the noload speed, stall torque, and inertia with a given gear ratio (N_g). The easy ones first:

$$\begin{aligned} \text{stall torque (at the catapult arm)} &= \text{stall torque (at the motor)} * N_g \\ \text{noload speed (at the catapult arm)} &= \text{noload speed (at the motor)} / N_g \end{aligned}$$

So, attach the catapult arm to a motor with a gearbox, and we get more torque, and less speed (assuming $N_g > 1$ and the output shaft spins slower). Makes perfect sense. Now let's talk inertia. Inertia is the resistance to rotating motion, analogous to mass for a linear system. The units for inertia are slug*feet², at least when using the foot-pound-second system of units (which for better or worse we will use). For a mass (in this case the boulder) on the end of a stick (catapult arm) the formula for inertia is:

$$J = \text{mass} * \text{radius}^2$$

Therefore, the inertia of the arm with the boulder is $0.02 \cdot 2.33^2 = 0.1086 \text{ slug} \cdot \text{feet}^2$. We are missing something however, the mass of the catapult arm. The heavier it is, the more energy we need, because we need to accelerate that as well (that's why it pays to keep it light). Our catapult arm weighs about 1.5 pounds, or 0.0466 slugs. The problem is that since all the mass isn't at the end of the arm, the previous formula for inertia is incorrect. We will use the formula for a uniform arm of the same length as the catapult radius, which is:

$$J = 1/3 \cdot \text{mass} \cdot \text{radius}^2$$

So we add an additional $1/3 \cdot 0.1086 \cdot 2.33^2 = 0.084 \text{ slug} \cdot \text{feet}^2$, bringing our total inertia to $0.193 \text{ slug} \cdot \text{feet}^2$. Now we have all we need to determine the initial acceleration of the catapult arm:

$$\begin{aligned} \text{torque} &= \text{motor stall torque} \cdot N_g \cdot (\text{motor no-load speed} / N_g - \text{arm speed}) / (\text{motor no-load speed} / N_g) \\ \alpha &= \text{torque} / \text{inertia} \end{aligned}$$

The only problem remaining is how to turn angular acceleration into the velocity and position of the catapult arm. For that we will make a simple discrete simulation, and to accomplish that goal we will make a brief foray into calculus, don't worry, we won't be solving any difficult equations.

Calculus, Integrals, and the Iterative Simulation:

To keep this short and on track, I'll state simply the following: velocity is the integration of acceleration, and position is the integration of velocity. What's integration? First a simple example. If you drive at 60 miles per hour for $\frac{1}{2}$ hour, you'll go 30 miles, that's obvious, that's integration. Here's the math:

$$\begin{aligned} \text{delta time} &= 0.5 \text{ hours} \\ \text{distance} &= \text{last distance} + \text{last velocity} \cdot \text{delta time} \end{aligned}$$

That's it. For the mathematicians this is called Euler integration. For the first delta time, distance = $0 + 60 \cdot 0.5 = 30$ miles. For the second delta time, distance = $30 + 60 \cdot 0.5 = 60$ miles. And so on, and so on. We will use this technique to develop the simulation of the arm motion. The only difference is that we will use a small delta time, say 0.001 seconds. So now let's lay out all our equations, and see what they look like:

Constant parameters:

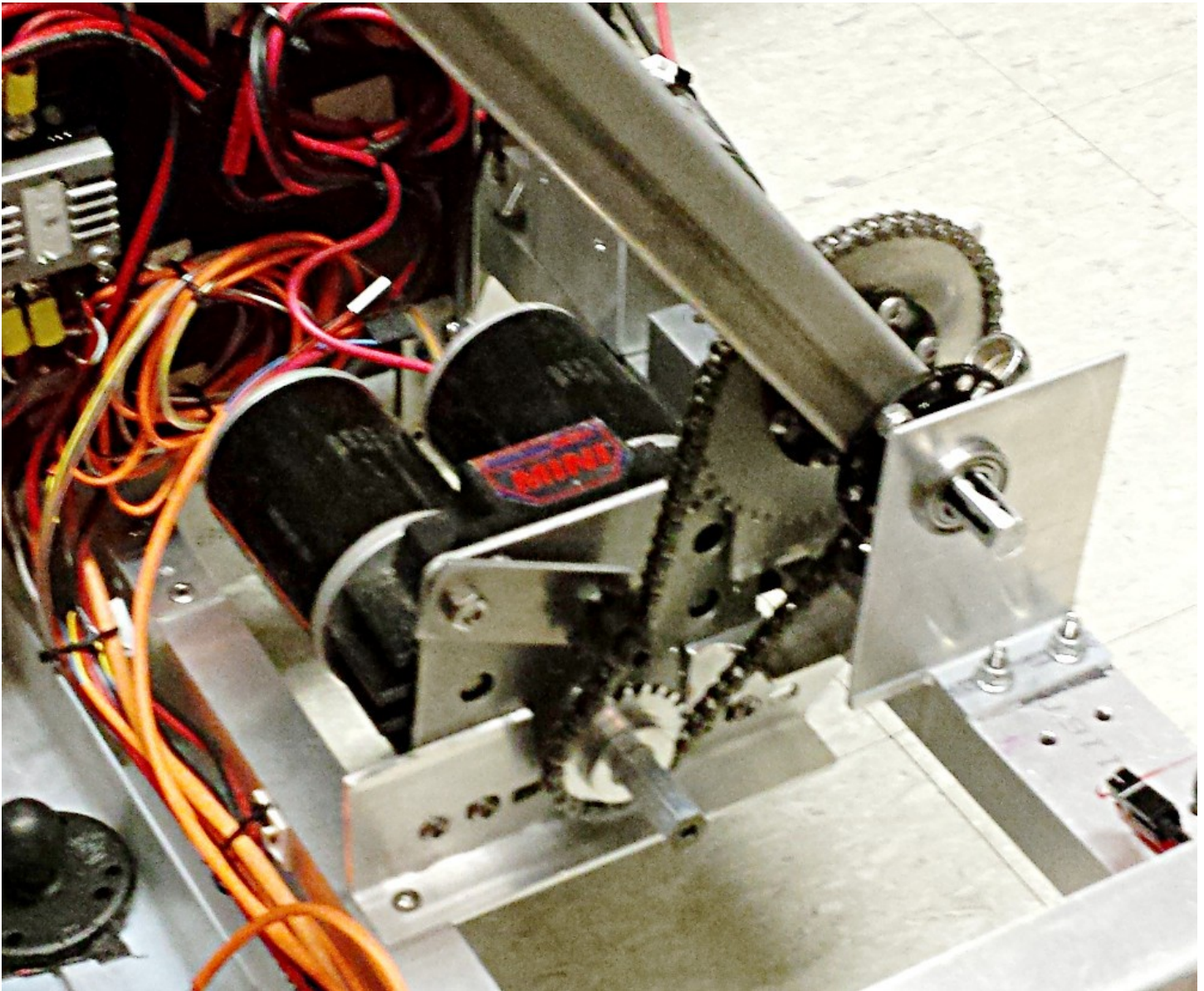
$$\begin{aligned} \text{motor stall torque (obtain from motor data sheet), for CIM motor} &= 343.4 \text{ oz} \cdot \text{in} / 12 / 16 = 1.79 \text{ ft} \cdot \text{pounds} \\ \text{motor no-load speed (obtain from motor data sheet), for CIM motor} &= 5310 \text{ rpm} \cdot 2 \cdot \pi / 60 = 556 \text{ rad/sec} \\ \text{radius} &= 2.33 \text{ feet (given)} \\ N_g &= 20 \text{ (pick one!)} \\ \text{arm stall torque} &= \text{motor stall torque} \cdot N_g \\ \text{arm no-load speed} &= \text{motor no-load speed} / N_g \\ \text{inertia} &= \text{ball mass} \cdot \text{arm radius}^2 + 1/3 \text{ arm mass} \cdot \text{arm radius}^2 = 0.193 \text{ slug} \cdot \text{feet}^2 \text{ (from above)} \\ \text{delta time} &= 0.001 \text{ sec (arbitrary, just pick something small)} \end{aligned}$$

Dynamic equations:

$$\begin{aligned} \text{torque} &= \text{arm stall torque} \cdot (\text{arm no-load speed} - \text{arm speed}) / (\text{arm no-load speed}) \\ \text{angular acceleration} &= \text{torque} / \text{inertia} \\ \text{arm speed} &= \text{last arm speed} + \text{acceleration} \cdot \text{delta time} \\ \text{arm position} &= \text{last arm position} + \text{arm speed} \cdot \text{delta time} \\ \text{boulder launch velocity} &= \text{arm speed} \cdot \text{arm radius} \end{aligned}$$

Note that I've glossed over some of the unit conversions, everything will work out if the units are functions of feet, pounds, slugs, and seconds (don't forget the radians!).

The only thing left to do is to put these into a spreadsheet, and simply vary the N_g parameter and look at the velocity when the arm gets to at 45 degrees ($\pi/4$ radians). It takes seconds to change N_g in the spreadsheet, and you can rapidly see what works best. Imagine trying 20 or 30 different gear ratios with real hardware. Could take a long time! You can also vary the power available with your motor selection. We used a toughbox mini to drive a #25 chain gear reduction, and we could use anywhere from 1 mini CIM to 2 full CIMs for power.



Prototype catapult drive train, FIRST Stronghold 2016

The following parameters were used to evaluate the various motor configurations:

Configuration	Stall Torque (foot*pounds)	Noload Speed (rad/sec)
1 CIM	1.79	556.0
2 CIMs	3.58	556.0
1 mini CIM	1.03	649.3
2 mini CIMs	2.06	649.3

Now that we have a basic simulation to exercise, we simply set up the desired motor parameters and play with Ng for that configuration. In the (hopefully) included spreadsheet “catapult_opt.xls”, we have 4 sheets, each with one of the 4 motor configurations above. Our hope was that a single CIM would have enough juice to get the boulder to the target, as we had plans for the 6th CIM (4 already in the drive-train). If it turned out that 2 full size CIMs couldn’t get the job done this would be a great time to know before spending a lot of time building a proof of concept demonstrator. After playing with the gear ratio for each configuration (using the starting range of 1 to 100 for lack of anything better) we rapidly converged on the following optimizations:

Configuration	Optimum Gear Ratio	Boulder Launch Speed (feet/sec)
1 CIM	25	32.9
2 CIMs	20	41.6
1 mini CIM	33	28.8
2 mini CIMs	26	36.3

Going back to our initial sizing requirements of 30 feet/sec minimum and 42 feet/sec desired launch velocity, we see that the single mini CIM case falls short, but the others meet the minimum required performance, with the 2 CIM configuration almost meeting the 200% energy margin requirement. At this point we tend to sharpen our pencils, and look to see if there isn’t any additional margin to be had. As it turns out, our catapult arm was more like -10 degrees in the full down position, giving us a little more room to accelerate, and the boulder leaves the robot about 1.5 feet off the ground, giving us more like 6 feet instead of 7 feet to clear to get to the high goal. At this point we have enough confidence to commit to building a prototype for test and evaluation.

Results:

Our initial prototype used 1 CIM motor with a 12.75:1 toughbox mini. We selected 22 and 48 tooth sprockets for an overall gear ratio of $12.75 \times 48 / 22 = 27.8:1$. We also bought a 60 tooth sprocket, to see if we could do a little better with a higher ratio. Initial testing looked OK, but we wanted a little extra margin to ensure we could attempt the longest shot on the field (corner just past the low bar). Since we wanted to preserve our CIM allocation for a potential climber, and our robot was light (really light, like 80 pounds) we went with 2 mini CIMs and the 48 tooth sprocket. Performance was good enough that we didn’t bother trying the 60 tooth sprocket since the smaller gear fit our packaging better.

Tools:

This example was developed around a spreadsheet application due to their wide spread availability and ease of use. There are however, other tools. The original optimization performed for the catapult was performed using a tool called Octave running on a Linux based computer. Octave is a MATLAB like tool that allows the user to write simple scripts to solve problem easily, and is my first stop for basic simulation. Last year we experimented by installing Linux and Octave on a students laptop, and used this environment to teach him basic simulation and control system principles with great success. An equivalent Octave program to the spreadsheet attached is included as an appendix for those who would like to try it out, it should also run under MATLAB with little to no changes.

Acknowledgements:

I’d like to thank Dean and Woody for coming up with such a great idea so many years ago, and the rest of the Team 230 Mentors who are busy doing the bulk of the real work required to run an FRC team while I sit and scratch my head.

Sincerely,
Steve Spoldi, FRC Team 230.

Appendix I Octave simulation example

The following is an octave program (script) that closely mirrors the calculations in the provided spreadsheet. This is my goto environment for this type of work, and should run on any Octave installation without change.

```
clear;

% constants
dt = 0.001;    % sample time between iterations
g = 32.2;      % gravity
r2d = 180/pi;  % conversion of radians to degrees

% standard cim motor
tstall_m = 343.4/12.0/16.0;    % motor stall torque, oz-in converted to ft-lbs
nlspeed_m = 5310.0*2.0*pi/60.0; % motor no load speed, RPM converted to rad/sec

% mini-cim
% tstall_m = 198.0/12.0/16.0;    % motor stall torque, oz-in converted to ft-lbs
% tstall_m = tstall_m*2;        % 2 motors
% nlspeed_m = 6200.0*2.0*pi/60.0; % motor no load speed, RPM converted to rad/sec

% catapult parameters
ng = 25.0;    % gear ratio
cat_length = 28.0/12; % length of arm in feet
cat_mass = 1.5/g;    % mass of arm in slugs
ball_mass = 0.65/g; % mass of ball in slugs
thetamax = 45.0/r2d; % maximum arm angle converted to radians

% derived catapult parameters
inertia = (ball_mass + cat_mass/3.0)*cat_length^2;
tstall = tstall_m*ng;
nlspeed = nlspeed_m/ng;

% initial conditions
theta = 0.0;
thetadot = 0;

% simulation loop
n = 2;
while (theta < thetamax)

    % calculate torque at the catapult arm based on current motor speed and ng
    torque = tstall*(nlspeed - thetadot(n - 1))/nlspeed;

    % integrate acceleration to compute angular velocity
    thetadot(n) = thetadot(n - 1) + torque/inertia*dt;

    % integrate angular velocity to compute angle
    theta(n) = theta(n - 1) + thetadot(n - 1)*dt;

    % next sample
    n = n + 1;
end;
```

```

% arm velocity
vel = interp1(theta, thetadot*cat_length, thetamax)
vx = vel*0.707;
vy = vel*0.707;

% performance metrics
max_height = vy^2/2/g
distance_at_max_height = 2*max_height

% plot data
% arm velocity
figure(1);
plot(theta*r2d, thetadot*cat_length, 'linewidth', 1.5);
title('Catapult Arm Velocity vs Position, ng = 25', 'fontsize', 12);
xlabel('Degrees');
ylabel('Feet/sec');
grid on;

```