Common project pitfalls and how to avoid them!

Based on previous experience with the final projects, I've seen a few pitfalls that I should address. When formulating your idea, you might think about whether you're at risk of any of these and how you could change your project to avoid them. This is for your sanity, not for my marking process!

1. Building lots of the same thing: In a hardware project, building the same thing over and over is very time-consuming and not very rewarding. If you're project begins with "I'm going to build a huge army of _____," then consider that just the time to sit there replicating the design over and over is likely to squeeze out the interesting bits! Projects like this work better on a longer timeline, where you can spend several months perfecting one or two items, then develop a mass-manufacturing process to make more. *Suggestion: build fewer of the item and make each one more interesting. Most projects are still worthwhile at smaller scale; if not, choose a different topic.*

2. Projects that are mainly mechanical: I've seen a number of proposals where the main challenges are all mechanical. Building robots or objects with lots of moving parts fall into this category. There's nothing wrong with mechanical stuff unless the *primary* challenge of the project is getting the mechanical pieces to stay together. Ask yourself whether the physical construction is more difficult than the wiring or the programming. If the answer is yes, then you might think about adjusting your idea. Unfortunately you'll find that mechanical bits are always the most failure-prone part of any project (I speak from experience) and you don't want the outcome to hinge too much on that aspect alone. *Suggestions: make the moving parts simpler and put more effort into the sensors and interaction. It's easier to sense motion than to create it. It's also easier to move things along one axis, and for short distances, than to move them around in multiple axes.*

3. The "trivial or impossible" problem: There are certain things that are both really useful and really hard to do. This can be hardware or software. If something is useful but hard, some company usually makes a pre-made widget to do it so others don't have to deal with it themselves. If your project falls in an area like this, you have a conundrum: either you use the premade widgets, in which case the problem is already solved, or try to replicate all the effort that went into making that widget (probably impossible in the timeframe). As a hypothetical example, suppose you wanted to develop your own skeleton-tracking library for Kinect. OpenNI already does it, and a project that just turns in OpenNI example code isn't going to be sufficient. But doing it yourself will be really complicated. Same idea with building your own consumer electronics; for example, making a touchscreen is really hard! *Suggestions: start by using the premade solutions, and ask yourself what kind of creative application you can build on top of it. The raw object itself is rarely as interesting as what you can build with it.*