So You Wanna Be A Roboteer?

Ideas & Observations

Michael J. Stigler

Owner & Engineer **silicis technologies inc**

While @ Beckman Coulter a coworker brought up an idea to me being implemented by the Rose-Hulman faculty, an interdisciplinary course structure to encourage an understanding of robotics amongst all engineers. I thought, "What a truly fantastic idea." I believe that there are many students across Electrical Engineering (EE), Mechanical Engineering (ME), & Software Engineering (SE) that could do very well in the world of robotics given the opportunity & exposure normally found only in high end projects or industry. The difficulty would be how to address the addition of necessary courses to give enough of an introduction to another discipline without interrupting ones current course load. Since I'm not versed in running an Engineering department at a top notch university I'll leave that to the experts. I will though try to offer up some of what I've experienced while TAing a Mechatronics course, conducting college interviews, my participation in two DARPA Grand Challenges, working as the only Engineer for a small military contractor in the world of UAVs & my own long running robotics projects which have given way to a robotics start-up.

Most of the deficiencies I've seen across disciplines have come from simply 'not knowing what they don't know' & 'not knowing what they do know'. Many engineers could do very well crossing over if someone were to simply tap them on the shoulder and for example say, "Hey, ME, here is your Rx interrupt, here are the bytes you Rx'd, and the two bytes are 0xBE and 0xEF." Most engineers will chew up a new challenge without giving it a second thought & suddenly they've become far more valuable to their team.

So, what areas do I believe everyone could use at least some exposure to and maybe why? I intend to convey many high level ideas that I could further elaborate on as needed. In no particular order, a list of 'key robotic skills':

1. MATLAB, a robotisist dream. With each release MATLAB grows into a more powerful platform for quickly testing concepts, honing algorithms & analyzing data. With knowledge of its rather simple language & the right toolboxes an Engineer can quickly perform a task such as the Kalman filtering of data which may have taken months without the tool. Other applications in control systems, machine vision & mechanics are made simple & fast to try. This tool when used correctly can literally save weeks & months during development. EEs & MEs typically have had exposure & some experience. Software Engineers typically haven't had exposure BUT would pound through the coding with ease. (To me MATLAB is synonymous with MATLAB, its various toolboxes, & Simulink.)
2. Oscilloscopes & Digital Multimeters, key tools in debugging any system. Absolutely essential ingredients to working on any robotics project. Deficiencies by EEs are inexcusable but often observed, those by MEs have been surprising to me, & most SEs seem to be able to pick the tools out of a line up. More work with these basic tools is needed as "I've used them in lab" just doesn't do much good after school.

3. ProE & SolidWorks, way better than a drafting board. Modern CAD/Modeling tools are indispensible & accelerate development. While MEs typically show some skill here it is very necessary for anyone engineering on a project to functionally use these tools. This will become apparent when a EE is perhaps designing a board for an area tucked back in a robot, or when a SE is working on kinematics algorithms with 5 degrees of freedom for a robotic arm.
4. Steppers and Servos, all motors are not created equal. There is an extreme deficiency here amongst graduating Engineers. Typically during an interview the answer regarding motors is *oh yea, we made stepper turn during lab.* It is **absolutely** imperative for any one on a project with motors (typically any robot) to understand the differences in the two motors and their respective subsets. I have seen this lack of understanding cost a lot of time in the long run because a motor has been mis-tasked by the designer & mis-used during implementation. SEs are not off the hook here, this goes to algorithms too, some motors just can't get to 10,000 RPM, not even if you continue to pulse it to do so. Steppers can be dependent on mounting angles and exhibit strange harmonics. Concepts like *linear, feedback, brushed & brushless* should be understood for servos. Deeper knowledge of motors will help all students looking to get into the field of robotics.
5. Datasheets, like a little biography about our silicon and metal friends. Datasheets, while often long, drawn out, potentially confusing, intense and dry, contain a wealth of key information. Understanding how to truly read a datasheet has typically been something I've seen come with experience & not a skill one has fresh out of school. The sooner any engineer figures out this art, if you will, they will start to save much time and money as they'll typically get it right the first time rather than the second or third or never. Also, during debugging and system development it is key for anyone involved to understand any part of the system they are working with and any subtleties mentioned by the manufacturer. Where could you find that information? Just one place, the datasheet. (This naturally extends to app notes provided by manufacturers.)
6. Sensors, giving our projects eyes, ears, and lasers. It is easy to hook up an IR rangefinder for lab. Sensors are typically straight forward with their connections, but to use most sensors correctly the joy ride typically ends there. Does a 16 bit A/D really give us 16 bits? How do we sample an analog sensor to really get a good representation of the output? If a sensor takes a reading every 10 mS & mulitplexes 8 nodes will it do the set in 80 mS? The output of that sensor is noisy, do we just buy a more expensive one? Our motor is clicking along at 45,000 RPM, surely this photo diode will tell us something? Surely all analog sensors give a very nicely calibrated linear output? This sensor is serial, my A/D won't work with it? Any engineer, whether designing a system or attempting to use data output needs to truly understand their tools. There are so many sensors available today; exposure to more & a variety of different units would be valuable.
7. Analog vs Digital, it's not as black and white as you'd think. I love digital, I really do. But as much as I'd like to deny it I live in an analog world, and I probably always will. So, my digital systems are affected by the pitfalls of the analog world as well.
    a. EEs coming out seem to have less knowledge of these issues. Knowledge of filtering, interference and design limitations due to analog are very important. Theoretical inputs and outputs of a system look great in the datasheet, but only exist in the real world in a Canadian trailer park with Elvis. Board design, signal routing, component placement, and design consideration all hinge on the Sinusoidals and Guassians of analog.

    b. MEs need to be aware during design not to put an extremely sensitive sensor board next to 36 volt, field inducing, RF emitting power supply. Wire routing, chassis materials and motors all drastically affect system performance.
    c. SEs, digital filtering can solve many problems not caught on the embedded board. Data carrying some noise is okay and can be handled but be aware. Work with your EE friends to make sure in the end your data represents the system as best you can. Nyquist was right ya know.
8. Embedded OR on a PC, where do you like your calculator. In designing a system it so obviously key for all parties to work closely together from the beginning. If the ME is aware that the EE will have some added time to use a DC Servo motor & the EE is aware the SE could easily schedule system events & the EE is aware you just can't make a XY stage stop on a dime the world will be a better place. Designing not only for yourself, but the entire system is a friction point I've often seen in industry. Perhaps a small sacrifice as a EE could save weeks for the SE on my team. Or, if my ME pal could free up an extra 9 cubic inches and we can double the size of the stepper & I will finish the embedded system in a week rather than 2 months for an alternate implementation.
9. Visual Studio, it's a neat place to edit icons? SEs have a powerful playground to work their magic. EEs who have mastered firmware can many times cross the bridge with minimal snags. Although the EE may never abstract all his business objects correctly, she'll sure have a far easier time running her machine vision algorithms for finding a test tube than she would on her 8-bit PIC. MEs, as always, while testing and debugging their part of the system can sure benefit from the knowledge setting a break point or changing some parameters in test app.
10. Soldering, not only a way to make jewelry and cheap Christmas presents for your Mom. Across the board, all engineers lack this skill fresh out of their University training. Hands down. They're like monkeys with two left hands. Please, please, learn how to solder correctly. (Let it be noted that just because you have solder, a soldering iron, a handful of components & a board you may not be *soldering*, do it right) Also, with places like 4pcb.com & student specials there is no excuse any more for a bread board & 47' of hacked up network cable. This skill alone has saved me so much time and money on every project I've every worked on.
11. Capture and Layout, more than what you do with the flag or at the beach. Board schematic skills are a must for a EE & laying out simple projects can save weeks. If this skill can be picked up at any time it drastically increases a EEs value. And if we've all learned to solder suddenly you're getting a prototype three weeks earlier for $1/10^{th}$ the price. Any engineer can benefit from at least knowing their way around the tool.
12. Control systems, it makes your air conditioner work. At a minimum the knowledge of a PID loop is necessary. But, for a EE, SE & ME the practical knowledge of actually implementing a control loop is lacking. It is very different to code for a loop than to do a root-locus plot. EEs need this knowledge & depending on how far up the control of the system is pushed this can fall squarely in the SEs lap. An ME isn't off the hook either; backlash, static friction and gear ratios all affect how the system will respond & thus how it will be controlled.
13. C, yes, there was something before the ++ version. The embedded world is a great place SEs, come on over. There are many times that doing something in an embedded environment just makes more sense, if an SE can just change hats the project will just keep going. It also can help when the SE understands what the embedded system is doing and how to design their software

to maximize the lower level embedded features. An ME that can poke through an IDE and lightly modify existing code is **FAR** more valuable than one who thinks *char* is a misspelling of their ride to work.

14. Materials & tolerances, balsa wood, although cheap & readily available, does not make for a good industrial robot frame. High level competence of the mechanics and design of a system by the other engineers will allow designs to happen more quickly. The physical limitations of a system are a reality, and there will be times that the EE may not get the 15 sensors in 1 square inch & if they're aware of this out of the gate all parties will benefit.
15. Embedded systems, not as easy as 1, 10, 11. Especially important to the EEs of the world. An embedded project in lab is far different from system running a full blown robot. When one is using 20 peripherals, including 4 forms for COMMs and sampling 16 sensors, timing is far more difficult than one ultrasonic range finder and a stepper. Real world examples of 'hard core' implementations could benefit the student engineer, even if it's over a few lectures. These are also considerations that should be made by the entire team. If 512 sensors are needed it's going to take more than 2 18F4680s and a MAX232.
16. Serial, it's not just for breakfast. RS232, I2C, SPI, CAN, RS485, 802.11.15, whoa. EEs in particular need to understand their system will probably have to interact with other systems. There are many protocols and standards available, all with their pros and cons. At one time or another, an embedded robotics developer will have to use all of them. The sooner they start to understand the differences and implementations the better off they'll be.
17. The Schooner, it once was a great way to travel. My final point & it goes to all engineers in robotics. Be cutting edge, embrace change, don't cling to old implementations, or be a one trick pony. I am by no means implying one should run rampant changing systems with no rhyme or reason BUT the world of robotics is changing at a rapid rate. Over just the past 5 years advancements in technology have made certain things possible that have never been considered before. Dedicated ICs are showing up every day from the major manufacturers that will make designs smaller, cheaper, more reliable and all around better. Hybrid systems that combine a processor core with a DSP now allow for an embedded system that previously required a small computer and a couple motor boards. Sensors get smaller, faster and more powerful all the time. Don't be afraid to utilize the new tools available & keep up with everything going on around you. Join IEEE & subsequently the Robotics and Automation group, go to robotic expos & grab a few of your dorm pals and participate in the Trinity College Fire Fighting or ION Robotic Lawn Mower Competitions.

The world of robotics is **extremely** difficult, challenging and frustrating. It is definitely not for all engineers. Now, and even more so in the future, robotics projects will be hands down the most difficult thing you'll ever attempt. But. But, after hundreds of grueling hours. But, after countless months of all nighters. But, after the first time you reach over and the robotic arm you just finished reaches back and gives you a *high five* you'll realize that for some there is no more rewarding experience in the world. Good luck. – Stig

Questions or comments? It'd be great to hear them: roboteer@silicis.com