A Twelve-Step Program to More Efficient Robotics

David P. Miller

■ Sensor abuse is a serious and debilitating condition. However, one must remember that it is a disease, not a crime.[1] As such, it can be treated. This article presents a case study in sensor abuse. This particular subject was lucky enough to pull himself out of his pitiful condition, but others are not so lucky. The article also describes a 12-step behavior-modification program modeled on this and other successful case studies.

Sensor abuse is a serious and debilitating condition. However, one must remember that it is a disease, not a crime.[1] As such, it can be treated. This article presents a case study in sensor abuse. This particular subject was lucky enough to pull himself out of his pitiful condition, but others are not so lucky. The article also describes a 12-step behavior-modification program modeled on this and other successful case studies. I hope that it is of use to researchers who suffer from this terrible affliction.

A Case Study in Sensor Abuse

Here is the story of a robotics researcher whom I call DM. DM was a theoretical planning researcher who wanted to work with real robots. He got his chance in 1988 when he took a job in California. There, he worked on several mobile robot systems, all with the goal of achieving rough terrain navigation. As can be seen from DM's story, researchers in this field are highly susceptible to sensor pushers; however, they can still free themselves from the bond of unnecessary sensing and eventually lead healthy and productive lives. Here, in his own words, is DM's story.

Mainlining Stereo

When I first got to California, I was sort of naive. I had heard stories about what life was like in Los Angeles, but I didn't really believe them. When I went into work, everyone seemed normal. Even the system we were going to use seemed ordinary—a fairly traditional architecture, a standard sense-model-plan-act cycle augmented with expectation-generation and execution-monitoring steps. I thought this would turn out to be an ordinary project, but thinking this way turned out to be my undoing, especially when the sensor man showed up pushing his wares.

I wasn't sure what the proper sensor modalities should be (my only previous experience had been with sonar), so when the sensor man suggested we use stereo imagery to build a detailed topographic map of the robot's surroundings, I agreed. Many of my friends had used stereo, and frankly, I was curious to see what it was really like.[2] We mounted cameras on a pan-tilt head, stuck it on our robot, and off we went.

The images were carefully correlated to a fraction of a pixel. The generated depth data were converted to plan view height data and stored in a set of 3 grids, each placed at one quarter of the grid-cell size and extended to twice the range of the previous grid. The lowest-resolution grid was at approximately 1-meter cell size and was extended to 32 meters in all directions from the robot. The smallest grid was at 6-centimeter resolution and was extended for 8 meters in all directions. The grids contained height, slope, and roughness estimates for each cell as well as confidence values for the data in the cell—over 100,000 cells in the layers and half a dozen fields to 1 cell. These numbers were great because we were producing a lot of data—megabytes! Even though we were doing stereo day and night, it looked like our productivity was way up...we had reams of data. It got to the point where I could do stereo right in front of my boss, and he didn't care. Sometimes, I'd be doing stereo for hours, just staring at a single set of images, but it was OK because it looked like I was productive.

Our system created a global map grid at 1-meter resolution before the start of the run, using stereo aerial imagery. Sure this method was expensive, and at first I had my doubts, but the sensor man told me I needed aerial stereo to get the best effect. He said it was a more natural "high." The global map was approximately 200 meters on a side. The local grids were matched into the global map. A gyro compass on board the robot removed any rotational uncertainty. A position estimate allowed the match to be approximated; a least squares fit tuned the match and allowed the robot's position to be calculated more precisely.

Through prior analysis of the global map, we assigned a local goal that lay along the edge of the local terrain data. A variety of path-planning techniques were used to find a path to the local goal; the path was then simulated in hideous detail using a quasi-static vehicle model. Quasi-static was more than adequate because between the stereo, the terrain matching, the path planning, and the simulation, we were working hard but not getting anywhere too quickly.

In this architecture, the perception system was king. It had to accurately model the local topography, match 2 representations of the topography whose data were gathered by different systems from different attitudes, and analyze the terrain with regard to how a vehicle would traverse it. This system was both computation and data intensive. It required over 10 megabytes of storage to compute and maintain the various terrain grids. Using a 2–million instruction per second processor for all processing, the system was unable to move at more than 2 meters an hour. Terrain matching was not very reliable because in benign terrain, there were insufficient features to get an accurate match. However, we didn't care if the terrain matcher really matched the terrain; after months of doing stereo, we had convinced ourselves that the important thing was to keep the variance in the correlation low. The depth maps were good; they had to be good—the numbers said so. Reality just didn't matter anymore. However, in rugged terrain, the robot's eye view was so heavily shadowed by near obstacles that the terrain looked different than it did in the aerial images. The generation of execution expectations was similarly hampered. To top it off, the camera could not see the robot's initial position or the area immediately in front of it. The robot thought it was standing on pedestals, one under each wheel. We practically had to talk it down—like a jumper on a ledge—to get it to start moving.

Stereo Laced with Speed and Its Effects on One's Ability to Navigate

After these initial experiences, I was tempted to try something radically different, but then the sensor man came back and asked how I liked doing stereo. When I told him my problems, he asked how much cash I had. When I told him, he said all my problems could be solved by the proper application of speed and a new outlook on how to get from here to there. After haggling for a while on price, I went back to the lab to test my new purchases.

With our new, fast stereo, we didn't really have time to sit around thinking about the world much; so, we simplified our model somewhat. We modeled obstacles and passive terrain but no details in either. After months of staring at monitors and doing stereo, our outlook was fairly black and white anyway. We also didn't have time to plan clever paths; so, we used a NAT (navigation template) algorithm to decide how to go around the obstacles. (After months of free correlating, the walls often seemed to be crawling with things such as gnats). Under the NAT navigation system, the space around the robot was represented by a flat, Cartesian plane. The local goal (usually on the order of a hundred meters away) was a sink in a gradient field spread over the plane. Obstacles were represented by spun repulsive gradient fields placed at their proper X-Y coordinates. The obstacle gradients were spun clockwise or counterclockwise depending on which direction the planner decided the robot should go around the obstacle.

The trajectory calculation was fast (approximately a second), and the NATs could be respun in about the same time. With the speed I had bought,[3] we were able to perform the stereo matching and obstacle filtering in about 30 seconds. The robot moved at a top speed of 2 meters a minute. Therefore, we were able to update the robot's direction of travel about every meter. My boss was upset at first that we had stopped producing reams of data. Then, my boss was happy because we were moving an order of magnitude faster. Then, my boss was upset when he realized this speed was still barely perceptible.

The speed helped a lot. I knew I would never be able to do stereo without it again. However, it severely colored my (stereo) perception of the world. The world became either traversable or not. There were no more slopes or surface

Reality just didn't matter anymore…

There I was—broke and about to lose my job and not a robot to call my own.

details. The terrain was all black or white, and I started to feel like there should be more.

I complained to the sensor man. He came back and sold me a stereo diagnostic system that displayed different colors as a function of the depth calculated for each pixel. I admit I spent hours doing stereo and speed and mumbling "the colors, the colors." It was a low point for me.

Perhaps, it was that the robot performance, despite the speed, was still questionable. Perhaps, it was that the robot could barely carry the computers and power supply. Perhaps, it was because I couldn't stop watching the colors long enough to return phone calls. When I didn't respond to our sponsor's phone calls, he eventually pulled the funding.

Although at first I thought losing our funding was a disaster, it turned out to be the best possible thing for me. Without funding, the sensor man had no interest in me. He reclaimed his cameras and vision boards. The creditors eventually even took the mobility system. Finally, they reclaimed that big color screen, and the sensor man lost his grip on me.

Total Reactivity: Why Do You Think They Call It Dopey?

There I was—broke and about to lose my job and not a robot to call my own. I went to the local bar to drown my sorrows. On the bar TV, I saw the story that saved my life. It was on one of those tabloid TV shows; some Australian Zen master was being interviewed. He was saying, "The world is not the way you think it is; the world is the way you sense it is." I had had a lot to drink, enough so that his statement made sense. Suddenly, I realized that if I used simple sensors, then the world would be simple too!

I went back to the lab, bottle in hand, and got the group together. We quickly built an indoor robot system that used only two bytes of memory to model the world. This robot was programmed to patrol an area, locate any objects it could pick up (Styrofoam cups), and drop the objects at a central location. The robot had a set of contact and proximity sensors. These sensors were wired to the robot's effectors using the ALFA (a language for actions) behavior-control language.[4] When a sensor fired, an appropriate behavior would be executed (that is, turn right if the left proximity sensor goes off). It worked—the robot did marvelous things and did them fast, but occasionally, it would get into loops, going back and forth between two obstacles. Somebody suggested that we add sensors to see if the robot was stuck in the same place. I fired him on the spot. Someone else suggested modeling the world and analyzing the robot's action using a rule-based expert system. My lawyer advised me not to tell you what we did with this person, at least not until after the trial. Finally, someone suggested making opposite behaviors asymmetric. In other words, a left-forward turn was at a different circumference than a left-rearward turn or a right turn. It worked! The robot sometimes looked like it wasn't firing on all registers, but it would always crab itself out of looplike situations.

In only a few days, we had a robot that would patrol a room, avoid obstacles, find lightweight small objects (like Styrofoam cups), and drop them near a lightbulb. Everything was fine except for the idea of the cup collection. My boss just could not understand cups. "Cups, Cups! Who cares about cups!?! Where this robot is supposed to go, there are no cups!" he said. When many other household items were suggested and rejected, I finally suggested dirt. He liked that idea, and we went to work.

Life off the Street

We quickly found out that the problem with dirt is that it is often accompanied by rocks, sand, plants, and so on. These elements were no problem for our original robot: It was huge; it had to be to carry all the computers, generators, and so on. However, we were operating on a shoestring now. The biggest wheels we could find in our price range were five inches in diameter. Somebody came up with a clever way of arranging the wheels so that the robot could get around pretty well in the dirt, but even so, the terrain looked formidable. It appeared as if the robot's proximity sensors would always be going off, causing the robot to wobble uncontrollably. The sensor man came by and said he had some range sensors—much more selective than proximity sensors—that he would sell us for cheap. I was almost sucked back in when I remembered the words of the Australian Zen master. Rip off the proximity sensors I told my chief engineer. "I canna do that captain, we'll crash for sure!" After much consoling, I finally got him to accede to my demands. When the proximity sensors came off, the robot lost its fear. The world was as it sensed it, and it did not sense much.

Off it went, armed only with *proprioceptive* (articulation and orientation) sensors. The robot passed perilously close to hazard after hazard. Finally, it ran right against a big rock, so big that it would surely flip the robot if the robot climbed it. It started to climb, and my finger inched toward the kill switch. It climbed more, and I was about to press the button when I realized it didn't matter if the robot flipped. This thing had an 8-bit processor, almost no memory, and almost no sensors. It wasn't worth anything. There was nothing to break, and even if it did, I could replace the part at a hobby store for $2.00. Even this worry was groundless.

The robot didn't know it was climbing a rock, but it knew it was tilting. Shortly before it reached a critical angle, it backed off and went exploring elsewhere. After that, I knew no fear. The robot just worked. Sometimes, it looked like it was bouncing around between rocks like a pinball, but it always got out. Asymmetric functions and increasing slippage always broke the loops. The project was a huge success, and I haven't used a sensor that produces more than a byte of data in over six months. Occasionally, I get the urge for a terrain map, but it passes. I'll never be completely cured, but I can control my sensing to a reasonable level.

The Twelve Steps

I hope you found DM's inspirational tale of use. Figure 1 shows the important steps to remember if you find yourself sensing uncontrollably. Remember, the complexity of the sensing system drives the complexity of the robot's planning and navigation system. The amount of planning and, therefore, the necessary computation, power, and vehicle speed are in large part dependent on what type of sensor processing is being performed and what information is extracted. Until proper legislation is enacted, the care and responsibility of your robot is left entirely to you. Researchers who abuse sensors are 10 times more likely to have brain-damaged robots than researchers who don't. Only you can stop sensor abuse.

1. Admit you have a problem. Don't be afraid, almost everyone does.
2. Analyze the robot's real goals. Is the robot's goal to build a map and use it to navigate to the office next door or is it just to get to the office next door?
3. Question your need for data. Are the data necessary for your algorithm or necessary to generate figures for your paper?
4. Have the robot remember what it has done—but not for long. Motor history can provide much useful information for your robot and really is simple to gather, but don't become an internal-state abuser!
5. Solve problems mechanically. Some sensing and control problems can be avoided by putting new shocks on your robot, but beware of the monster truck syndrome.
6. Ask if this sensor is the right one for you. If all you want to know is whether there is a big obstacle in front of you, stereo vision might not be the best idea.
7. Don't be afraid to experiment. Sensors don't always deliver what you think they will; test them out.
8. Avoid simulations. They are bound to succeed.
9. Avoid color monitors. They are bound to distract you.
10. Don't be afraid to have your robot ask questions. There are no stupid questions, only stupid robots.
11. Never forget that you have a problem. Just because the price on the camera system has come down does not mean that you have to have it.
12. Acknowledge that there is a higher power; it controls your funding; and it really doesn't care how your robot does something (and usually doesn't care what this something is), just as long as it does it.

Figure 1. The 12 Steps to Sensor Control.

Acknowledgments

I want to thank the National Center for Disease Control. After several unsuccessful attempts at getting them to recognize sensor abuse as a serious affliction that could potentially spread and was certainly worthy of future research, it finally offered some useful advice. The suggestion to take a long nature walk[5] was just what I needed to clarify my thoughts and set them down on paper. I also want to thank DM for the open and forthright telling of his story. Finally, I would like to thank his many colleagues who contributed to this work, reviewed earlier drafts of this article, and made useful suggestions. Like DM, they prefer to remain anonymous.

Notes

1. It might never be considered a crime unless we get some competent roboticists into the United States Congress.
2. I knew there were risks, but what the hell, I grew up in the sixties.
3. It was 100-percent pure Data Cube boards.
4. It was the first language we could get to.
5. *Take a hike* was the center's exact language.

Dave Miller spent four years at Yale University where he got his doctorate in computer science, married a clinical psychology student, and attended a great many parties where everyone's keys were confiscated till the next morning. All these experiences led to his interest in 12-step programs. He then spent the next 8 years flitting from coast to coast, working as an assistant professor at Virginia Polytechnic Institute and State University, a staff scientist at the Jet Propulsion Laboratory (where he met DM), and a visiting scientist at the Massachusetts Institute of Technology. His interests include studying robot navigation and minimal models of intelligence and finding steady work.