THE NEUROMORPHIC ENGINEER
A PUBLICATION OF INE-WEB.ORG

10.2417/1200902.1523

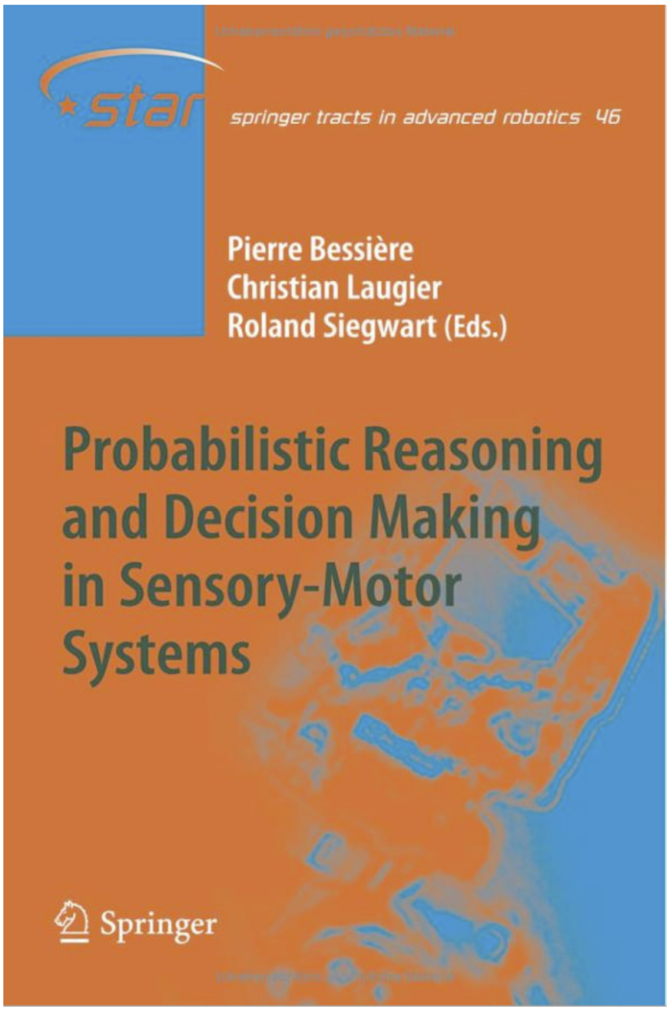
BOOK REVIEW

Probabilistic reasoning and decision making in sensory-motor systems

Matthew Cook

A dozen robotics-related PhD projects are brought together to provide a good overview of the state of the art in solving problems by defining joint probability distributions over both sensor readings and actions.

Pierre Bessière, Christian Laugier, Roland Siegwart (Eds.)
Publisher: Springer
Hardcover: 2008, 378pp, $149.00 **ISBN:** 978-3-540-79006-8


star
springer tracts in advanced robotics 46
Pierre Bessière
Christian Laugier
Roland Siegwart (Eds.)
Probabilistic Reasoning and Decision Making in Sensory-Motor Systems
Springer

It may come as a surprise that there is not a large literature on the use of probabilistic methods for robot control. After all, in the last 20 years there has been a huge amount of research into probabilistic methods in many areas of artificial intelligence, even in communities that are more neurally-focused than mainstream AI. However, almost all of this research has been limited to the sensory side: given some noisy observations or a noisy data set, what should we conclude? To complete the input-output loop, as is the purpose of any nervous system, we need to generate concrete motor commands. Are probabilistic methods of any use for this? For those of us who stay up at night wondering about this question, this book is clearly not to be missed.

Probabilistic Reasoning and Decision Making in Sensory-Motor Systems describes a dozen robotics-related PhD projects that were completed using the Pro-BT^{TM} probability calculation software, developed in Pierre Bessière's group, which is free for research and teaching use. However, this software is not needed in order to understand the methods used by the projects. These projects provide a good overview of the state of the art of this approach, in which problem solutions are encoded as joint probability distributions over both sensor readings and actions.

The first two chapters present the foundation used by the twelve projects described in the remaining chapters, which

Continued on next page

appear to have been written by the PhD students who completed did the research in each case. Each of the latter chapters therefore stands on its own. The book is not structured such that a body of knowledge is built up over the course of the book, so after the first two chapters, the order of the remaining twelve is rather arbitrary. They are grouped into six focused mainly on robotic navigation, followed by three industrial applications, and three psychophysical modeling applications. The average reader will probably want to read the first two chapters and then skip to whichever chapters seem the most relevant to their work.

The problems are described solved through 'Bayesian programs,' which are essentially the Pro-BT version of standard probabilistic graphical models, and they are always given in a textual form. The end of Chapter 2 briefly argues that Bayesian programs are superior to standard graphical models, due to their ability to represent joint marginal distributions and cleanly represent a system that includes many copies of a submodel.

The book shows by example how to write simple programs that use sensory inputs to produce motor outputs via carefully crafted 'probability distributions,' which I feel compelled to put in quotes because these functions are not really the true probability distribution of anything at all. Rather, the probability calculus formalism is used as a calculational tool that enables easy programming of the relationship between sensory input and motor output by simply describing the 'probabilistic' relationship between them.

For example, to program the idea of 'stop on red, go on green,' you could simply define a function $P(stop|red) = 0.9$, $P(go|red) = 0.1$, $P(stop|green) = 0.3$, $P(go|green) = 0.7$. Then at run time, if the robot noisily observes a light that seems to be red, say with $P(red) = 0.8$, $P(green) = 0.2$, then you can use the simple rules of probability to calculate $P(stop) = 0.9 \cdot 0.8 + 0.3 \cdot 0.2 = 0.78$ and $P(go) = 0.1 \cdot 0.8 + 0.7 \cdot 0.2 = 0.22$. Based on this estimate of the action, it is easy to define a simple rule for what to do, such as doing the action with the highest probability. Of course you could could at each instant take a random action with the given probabilities, but this has the undesired effect of continuously switching back and forth randomly between actions. Therefore, the maximum or average value is generally used for selecting the action, which means the original function P did not actually represent any particularly meaningful probability distribution, but was instead just a convenient way to write a simple program using four numbers describing the behavior in some sense. (The last of the robot navigation chapters does show one way to make the probability distributions be meaningful, at the cost of a significant increase in complexity.) This way of programming by defining probability tables is indeed a radically different way to program from traditional imperative languages.

Many of the projects have nice results in which the robot moves smoothly, like a biological system, rather than making sudden jerky transitions between actions as is often seen in traditional robotics. However, this smoothness has little to do with using a rigorous probabilistic approach, but is rather due simply to using a method in which it is easy to make smooth transitions between states, yielding smoothly transitioning behavior.

To illustrate this, suppose you want your robot to move forwards until it reaches a wall and then turn right. A traditional solution would be to move forwards at a constant speed s until the front wall sensor reading w exceeds some threshold w_0, at which point we stop moving forwards and start turning to the right with a constant rotation r. The resulting behavior exhibits the sort of unnatural constant-speed jerky movements that we commonly associate with the idea of 'moving like a robot.' Suppose on the other hand we use the front wall sensor as an analog control to switch smoothly between the two behaviors, so at any given time we move forwards with a speed of $s \cdot (w_0 - w)$, and at the same time we also turn to the right with rotation $r \cdot w$. Then as the robot starts to approach a wall, w will start to increase, and the robot will start to slow down and will start turning a bit to the right. The closer the wall is, the slower the robot will go and the more it will turn, and if turning to the right results in the obstacle no longer being sensed in the forward direction, the robot will start to increase its speed again. The resulting behavior looks much more natural than the traditional robotic behavior. Indeed, non-probabilistic methods such as fuzzy logic have been producing smooth behavior in just this kind of way for a long time.

The probabilistic approach used in this book also winds up handling this sort of problem in just the same way, since the linearity of probabilities ensures that as an observed variable changes linearly from one value (distribution) to another, the probability (distribution) estimate for any other variable will also change linearly. And of course the probability calculus also provides a smoothly varying solution for the target distribution even when things get nonlinear, due to the change in the input being nonlinear or involving multiple variables.

However, there are various loose ends with this probabilistic approach. If a non-smoothly varying output is desired, then this must be obtained either via non-smooth jumps in the input, or through a non-smooth conversion of the output distribution into a motor command (such as by using the maximum

Continued on next page

of the distribution). These solutions do not fit within the probabilistic framework, but rather into the pre- and post-processing steps, indicating that there is definitely more to these systems than just their probabilistic portion. Another issue confronting any Bayesian approach is how to pick the priors, but nowhere does the book shed any new light on this issue. It would also be nice to see how to structure larger programs in this framework. The last two navigation chapters give some hints about this, but this topic does not get explicitly discussed in its own right.

Of course, a key issue with using probabilities for motor control is the pesky issue of how to choose a motor command from a computed distribution of possible motor commands. Unfortunately this issue also remains an unaddressed shadowy art, as it only gets addressed by giving a simple list of the most obvious ideas in Chapter 3 (pick the max, pick the mean, etc.). The point that the probability distribution $P()$ is usually not meaningful in most of the examples is simply ignored, although the last of the navigation chapters puts some effort into trying to make sure it remains meaningful.

Even after reading the whole book, I do not see a clear methodology to how the solutions to the various projects were designed. Each appears to be an *ad hoc* solution, and from reading the descriptions, some were clearly built by adding hack on top of hack until they somewhat worked. If there were any projects that never really worked at all and were dropped, the book doesn't mention them or what their difficulties were.

It would have been nice to see some text discussing what sorts of overall lessons were learned from all these projects. Clearly these projects derive conceptually from the idea that a unified probabilistic approach can be applied to a variety of robotics applications. The contribution of the individual projects was then to show how this plays out in practice in a variety of contexts. Was anyone actually paying attention to all of the projects? Assuming somebody was, did they get any wiser by seeing what happened in the various projects? It would be very nice if that person would share their gained wisdom with the reader. I would have thought that this would be one of the main points of such a book, but unfortunately each chapter seems to have been written solely by the student working on that individual project, with little awareness of what was happening in other projects, and there is not even a final summary at the end of the book.

One of the advantages of having a book published through a large well-known publisher such as Springer should be that the final production process should be of high quality, but in fact this is one of the more poorly produced books I have seen. This book has never been proofread by anybody: the preface is full of typos, the main text sometimes has glaring errors such as raw LaTeX commands being visible due to typos in the source, and there is even a table with column headings in English and row headings in French! Many of the figures are of such poor quality that they are illegible. Even the binding of the book itself is of poor quality for the price being charged: the pages are glued, not sewn, and just by turning the pages as I read the book, the first 80 pages in my copy are now unattached for their bottom three centimeters. It is sad that Springer, famous for charging ever more exorbitant prices to university libraries, does not even bother to provide the most basic quality standards that readers and authors would expect from an established publisher.

All in all, if you are interested in how probabilistic methods can be used for robotic control problems, then I recommend this book for the wide range of examples it gives.

Author Information

Matthew Cook
Institute of Neuroinformatics
University of Zurich and ETH
Zurich, Switzerland
California Institute of Technology
Pasadena, CA
http://co2.ini.uzh.ch
http://www.ini.uzh.ch

Matthew Cook is a group leader at the Institute of Neuroinformatics in Zurich, and also holds a visiting associate faculty position at the Caltech. His research on cortical models of computation includes the application of probabilistic methods to robotic control.