On Why Better Robots Make it Harder

Tim Smithers

Euskal Herriko Unibertsitatea
Informatika Fakultatea
649 Postakutxa
20080 Donostia
Espaina

Universidad del País Vasco
Facultad de Informática
Apartado 649
20080 San Sebastián
España

Email: ccpsmsmt@si.ehu.es or smithers@si.ehu.es

Abstract

In this paper I discuss something which I believe is a common, though largely unreported, experience of people who build and use real robots, but which to people who don't, can seem counter intuitive: as we build better robots they become harder to use. I use this discussion to suggest that at least some of the difficullty is a result of thinking of robots as information processing systems, and of sensors as measuring devices, in particular. As an alternative, I suggest that viewing robots and their environments as agent-environment systems, whose interaction dynamics have to be got right, is a more appropriate approach to understanding adaptive behaviour in robots and animals.

1 Introduction

Quite often, when I explain to people, who do not build robots, how difficult it can be to make the small and relatively simple robots I work with behave reliably and robustly in ordinary environments (see [Donnett & Smithers, 91], [Nehmzow & Smithers, 91], and [Nehmzow & Smithers, 92], for example) they respond by saying "Why don't you use better sensors?" Or, "Why don't you build better robots?" A related feeling is sometimes expressed by people who do build robots, or at least have done in the past, when they say "I can't do what I want to do because I need a better robot." Or, "I've given up using real robots until the technology gets better." Yet a third kind of statement that is sometimes heard goes like this: "I don't want to have to deal with all the uncertainty and unrepeatability of real robots in my investigation of adaptive behaviour." Typically these last two are offered as justification for using so called robot simulations, and assume that the variations that do occur in real robot behaviour is something that could be made to go away, if we made better robots[1]. (Superscript numbers refer to notes at the end of the paper.)

I believe that this experience of better robots being harder to use, not easier, is a common, though largely unreported, experience amoungst people who build and use real robots, but which, as suggested above, is often counter intuitive to people who don't.

In this paper I attempt to show why the problem of making real robots behave reliably and robustly in real environments is not simply to do with the quality of the robots used, or with any inadequacy of the technology available. I do this in two parts. In the first part I present a story from the history of control and describe a problem that arose with the use of fly-ball governors as their manufacture was improved. In the second part, I use data obtained from a real robot operating in a real (though simple) environment to show that using higher resolution sensors introduces more variation into the sensor signals, not less, which in turn, can make achieving reliable and robust robot behaviour more difficult. I end by suggesting that this is a consequence of trying to understand and build robots as information processing systems, and that a more appropriate approach is to try to understand the interaction dynamics of these kinds of agent-environment systems.

2 Better Speed Regulators Worse Speed Regulation

The first widely used regulator was Jame's Watt's fly-ball governor, or centrifugal govenor[2], although many different regulation devices had been invented and used before (see [Mayr, 70]). This device was introduced in 1788 on the Boulton-Watt steam engines then built to power mills, and subsequently other factory machines. It's development and use coincided

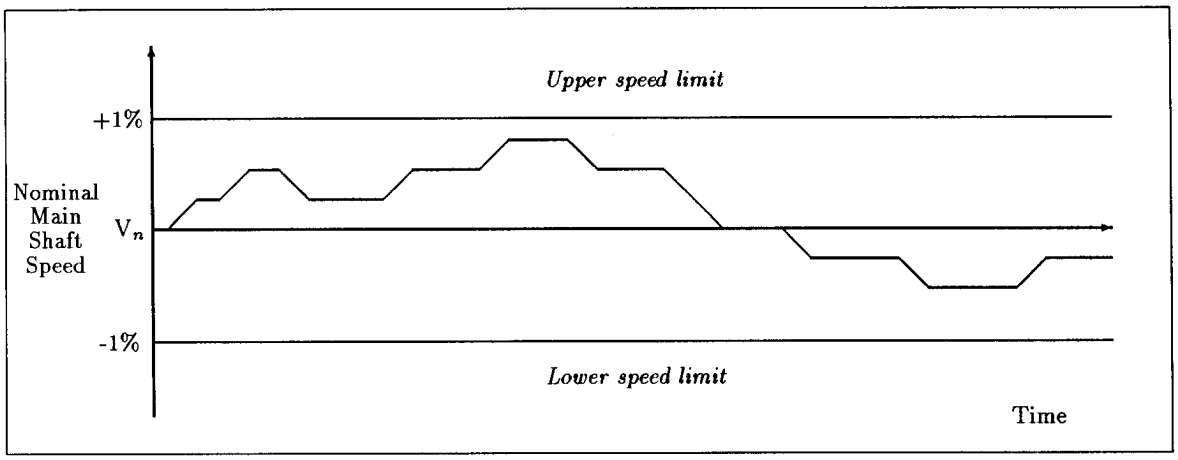

Figure 1: Speed variation in a well regulated main shaft as different machine tools are brought in and out of operation.

with the invention of an effective steam throttling valve originally intended for manual control, [Mayr, 70, page 111] and [Dickinson & Jenkins, 91, page 220], a necessary co-development to implement effective control using the fly-ball governor.

2.1 Speed Regulation in Early Machine Tools

Steam engines, and in particular those built by the Boulton-Watt company, became the main source of power in the factories of the industrial revolution in Britain. A typical arrangement involved a factory building with a steam engine installed at one end to supply power to a number of different machine tools distributed on the shop floor which occupied most of the remainer of the building. See [Kurzweil, 90, page 6] for a good illustration.

The power distribution consisted of a system of shafts, pullies, and belts. The steam engine was used to drive a main shaft running the length of the shop floor (usually some five or six meters above the ground). Each machine tool was then driven from this main shaft via a belt running down to a pully on the input shaft of the machine. To maintain a good quality of work, it was important that the speed at which each machine was driven remained constant, with only a small margin of variation. The Watt fly-ball govenor was used to "iron out" the otherwise large variations of speed which occurred due to the intermittent use of each machine: as different machines took power from the main shaft so the speed would drop, this resulted in a falling of the weights of the fly-ball governor which, in turn, resulted in a proportional openning of the steam throttle valve thus allowing more steam to the engine, and subsequently more power output on the main shaft, and a restoration to nearly the original speed[3], see figure 1.

At first, the use of the Watt fly-ball governor to regulate the speed of a set of machines tools powered by one steam engine worked well—the variation in the operating speed of each machine was kept well within acceptable limits. As manufacturing techniques improved the components of steam engines and, in particular, those for the Watt fly-ball governors, began to be made with increasing accuracy and to closer tolerances. However, rather than leading to an improvement in the operation of the machines tools—smoother and smaller speed variation—these better steam engines and, in particular, these better governors resulted in the appearance of a problem that had not occurred before. The so called *hunting problem.*

2.2 The Hunting Problem

The symptoms of the hunting problem, in this case of main shaft speed regulation, is a failure of the shaft speed to settle down to one steady speed but to continue to fluctuate up and down in a sine wave oscillation, see figure 2. This unsatisfactory behaviour is caused by the fly-ball governor first dropping thus openning the steam throttle value, which results in the engine speed increasing, but this time beyond where it is supposed to be. When the main shaft speed is too high, the fly-ball governor opens out and in turn closes the throttle value, thus reducing engine speed. In certain conditions this over correction for too slow a speed, resulting in too high a speed which is then over corrrected to produce too slow a speed again, can go on either forever, or for a long period of time before it finally dies out. When this happens the system is described as "hunting" for the correct speed.

The reason for hunting is a lack of *damping* or friction in the system. In the early speed regulated steam engines sufficient damping naturally

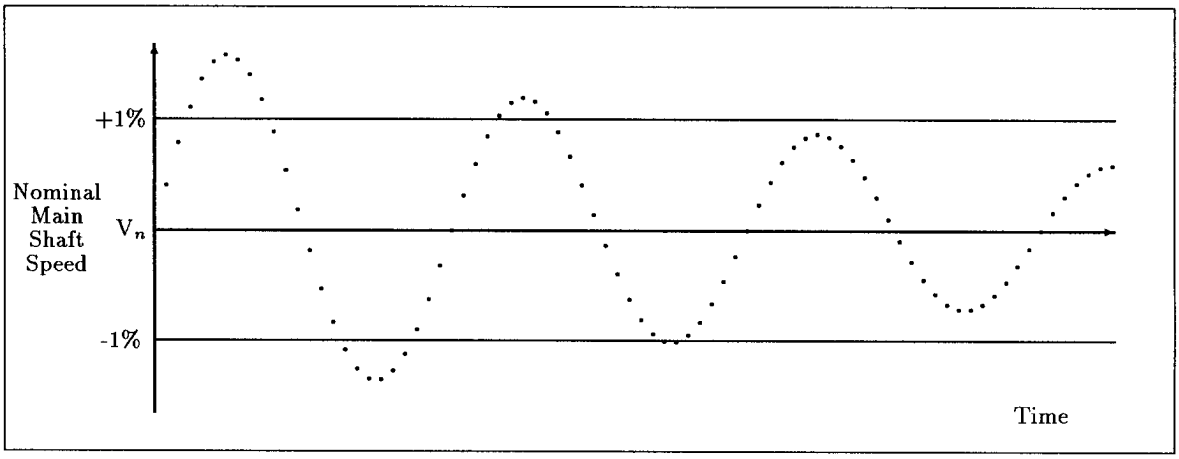

Figure 2: Hunting in an under-damped main shaft speed regulator.

existed as a result of the friction in the joints, bearings, and gears and/or pulleys of the mechanisms used. However, as the components became better made this friction was significantly reduced and the controlled system became seriously under-damped, and hunting began to be observed.

The hunting problem is not confined to the speed regulation of steam engine powered main shafts. It can occur, under certain conditions, in any feedback regulated or servo-controlled system. Today modern regulators and servomechanisms have additional components to prevent hunting from occuring under normal operating conditions—see proportional integrating derivative (PID) control in any control theory textbook[4]. In James Watt's day, however, this problem was not understood, and it became an increasingly serious one. It wasn't until a paper by James Clerk Maxwell, called "On Governors" published in 1868, [Maxwell, 68], that a mathematical treatment of these devices became available, and so enabled the hunting problem to be properly understood and effective solutions to it developed.

From this example of a problem that occurred early on in use of regulators and the beginnings of control theory, we can see that technological developments and improvements do not necessarily simply lead to better performing systems. It further illustrates that, as is typically the case, theoretical developments are what are needed to undertsand how better systems can be successfully designed and built, not just technological ones! Though modern controllers can now be used to achieve good regulation, where fly-ball govenros could not, they are more difficult to use—the gain values of the PID controllers, for example, have to be carefully set and adjusted.

We see, from this example, that though theoretical and technical developments have made it possible to design and build better regulators, they are not, in general, easier to construct and use. The same effect can, I think, be seen in other examples. It has tended to lead us to believe that the development and improvement of our theoretical and technical understanding, of how to build systems, makes their construction and operation easier. This, I believe, is an illusion: these advancements certainly enable us to design, build, and use systems that we were not previously able to construct, but they are not (in general) necessarily any easier to design and use for this. More things become possible, but they don't typically get simpler.

3 Better Sensors Worse Sensing

I now turn from an historical example to one based on data collected from a recent real robot experiment conducted at the VUB AI Lab in Brussels. This particular experiment forms part of an ongoing investigation into the dynamics of agent-environment interactions, [Smithers, 92] and [Smithers, 94], during which the sampled eight bit signals, from the five infra-red (IR) sensors on the robot used, were recorded. I first briefly describe the robot, the experiment design and setup, and data recording. I then present some of the results to illustrate the kinds of normal signal variation that can occur in this quite simple agent-environment interaction system.

3.1 The Robot and Experimental Setup

The robot used is a second generation Lego vehicle[5] (LvII). It has one-bit bumper and whisker sensors for contact detection, five eight-bit active IR sensors (all operating at the same IR and pulse frequencies) arranged in an arc from left to right across the front of the vehicle (with the centre one facing directly forward, inner side ones at 15° from forward, and outer side ones at 40° from forward), a revolution counter on the front wheel[6],

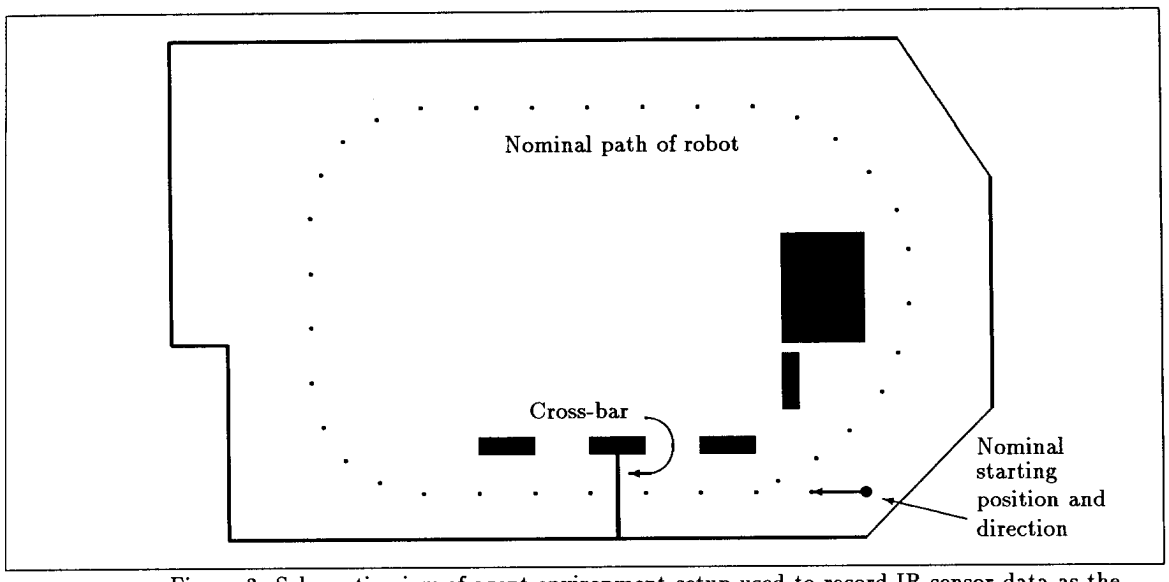

Figure 3: Schematic view of agent-environment setup used to record IR-sensor data as the LvII moves round the nominal path shown.

and a lap counter (see below). It is powered by two Lego 9v motors each supplied with 14v and controlled by a pulse-width modulation. This gives us more or less continuous variation of the power setting of each motor, and thus effective speed control and direction of the vehicle. It has a third free wheel at the front. (See [Donnett & Smithers, 91, figure 5] for an illustation of this arrangement.)

The LvII is programmed so that its default motion is to go forward in a straight line, and to use bumpers and whiskers for contact detection and IR sensors for object proximity detection in a 'don't get stuck' behaviour—do not get stuck in or trapped by things in the environment. The transfer functions, implemented by the program, between each sensor modality and motor states are dynamic (depending on recent sensor signal history), nonlinear, and independent of each other, i.e., there is no "sensor fusion" done in the program.

The experimental environment was formed by an enclosure and constructed to be a good "IR environment" so that the robot would normally not make contact with the sides of the enclosure or any of the objects within it, see figure 3. The parameters of the transfer functions, implemented by the control program, were adjusted so that, starting from the same nominal position and direction, the LvII would consistently take a route round the enclosure near the walls, passing under a cross-bar on each lap, thus triggering a sensor used to detect this event, see figure 3. Achieving this turned out to be significantly more difficult than it was for a first generation Lego vehicle (see [Donnett & Smithers, 91]) which used only three one-bit IR sensors.

Each run of the LvII consisted of ten lap counts and lasted a little over two minutes. During each run the signals from the five IR-sensors were sampled every 20 milliseconds and recorded in memory, together with the time and revolution count at the start of each lap. These data were uploaded to a host computer and stored at the end of each run. A series of nine runs were done, each of nine complete laps. The data presented here is taken from one of these runs (selected at random), with the data from the first lap disgarded to avoid transients associated with the starting conditions, giving a total of eight laps.

3.2 IR-Sensor Data

Space does not allow the data for all eight complete laps to be presented. Four sets have therefore been selected (again, at random) and are presented in figures 4, 5, 6, and 7.

As can be seen from figures 4 to 7, while there is an obvious common general form, there is considerable variation in detail of the five sensor signal profiles across each of the laps shown. What is important to note here is that this detailed variation is *not* due to noise. There is, of course, noise in these signals, but it can be shown to be at least an order of magnitude smaller than the variations that can be seen here. This variation is structural and arises as a direct result of the small variations in the actual path taken by the robot on successive laps. The lap times and distances travelled, and their percentage of the average values over the eight laps (see figure captions) indicate that this variation in path is small. However, it has a large (structural) effect on the actual sensor signals generated, as can be seen.

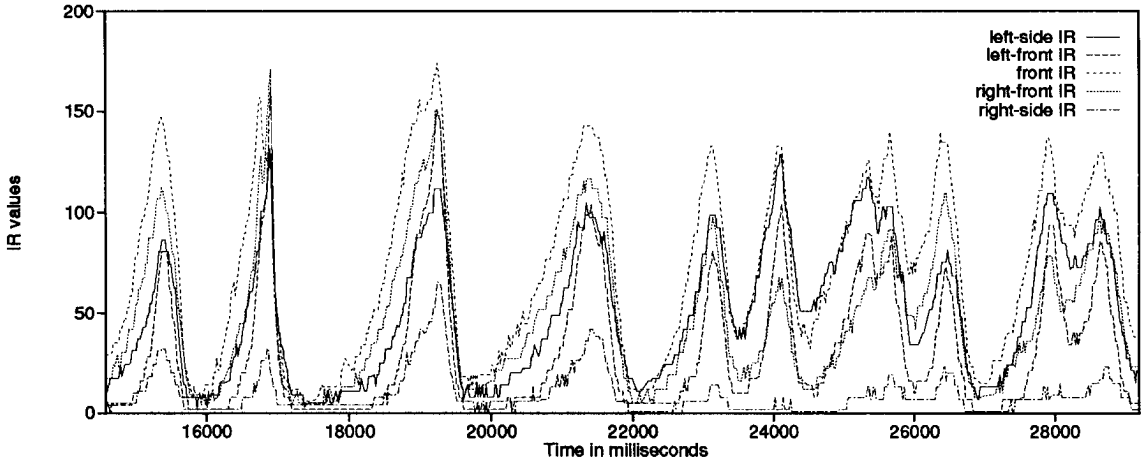

Figure 4: Lap 1/8, time=14593ms (101.48% of average), count=219, giving a distance of 9.862m (101.45% of average), and an average speed of 0.676m/s.

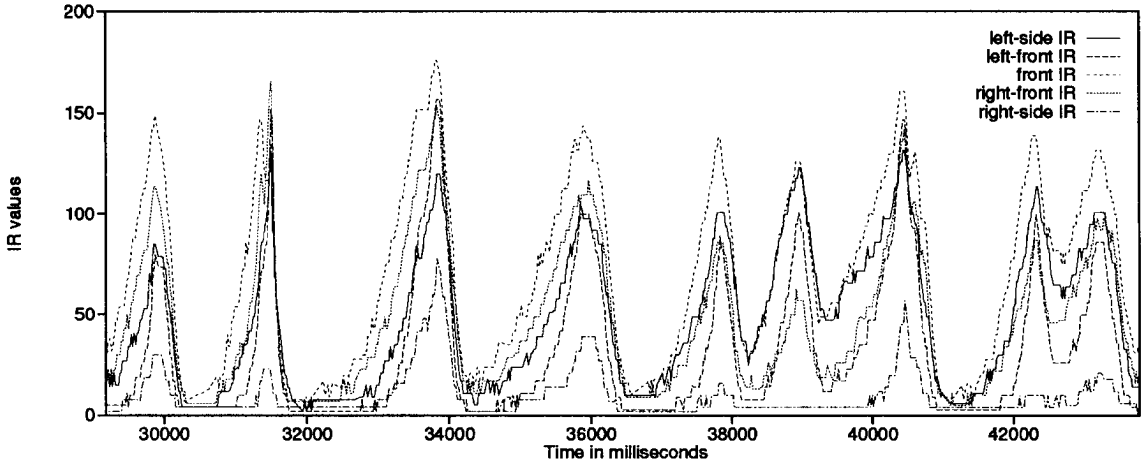

Figure 5: Lap 2/8, time=14595ms (101.49% of average), count=219, giving a distance of 9.862m (101.48% of average), and an average speed of 0.676m/s.

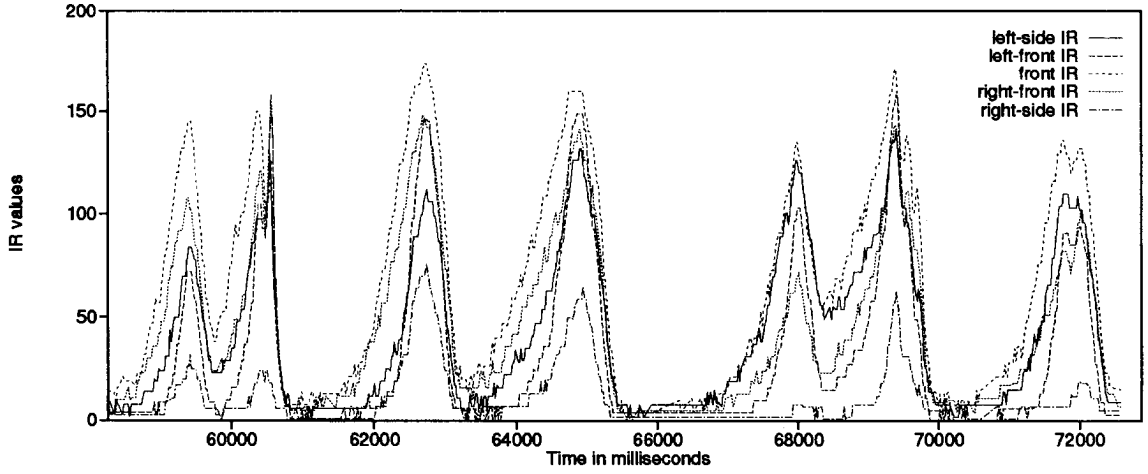

Figure 6: Lap 4/8, time=14293ms (99.39% of average), count=215, giving a distance of 9.682m (99.60% of average), and an average speed of 0.677m/s.

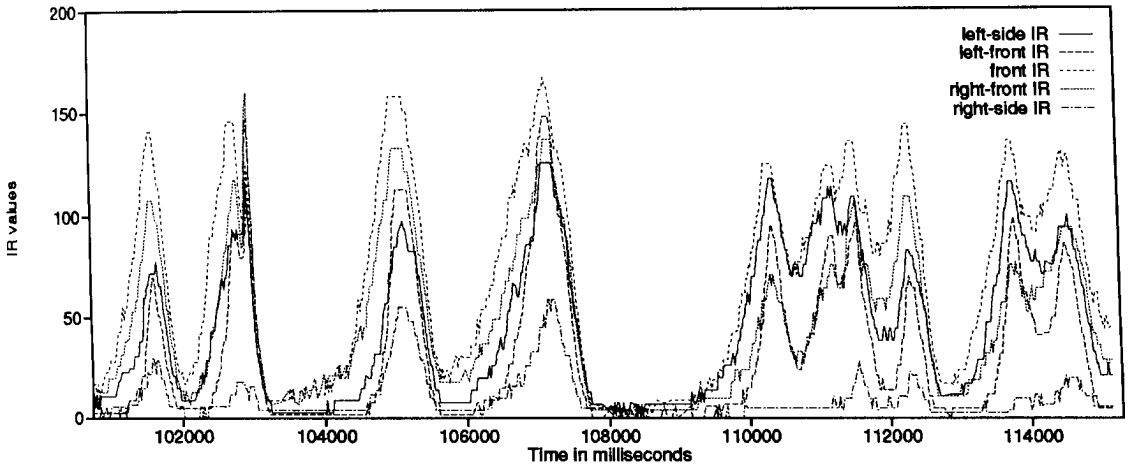

Figure 7: Lap 7/8, time=14459ms (100.55% of average), count=216, giving a distance of 9.727m (100.06% of average), and an average speed of 0.673m/s.

4 Discussion

The kind of structural variation in sensor signal profiles presented here is typical of other types of sensors used on robots, ultra-sound, and vision, for example. It can, of course, be reduced, and, at least sometimes, made effectively to go away, but only at the cost of controlling the motion of the robot so that the differences in the path taken (on each lap, in this case) become very small. This is both a difficult thing to do, and is not necessary for effective robot behaviour—the behaviour achieved is quite stable and not overly sensitive to perturbations. In other words, to reduce the sensor signal variation to a level that no longer matters—to make all the profiles effectively the same—requires much higher degrees of motion control than is necessary for the robot to get around quite well enough. This is the case in the experiment described here: the actual path taken is sufficiently close to the nominal path on each lap and sufficiently robust. To try to reduce what variation there is, to make the sensor signal profiles more similar for each lap, would take us away from the real problem of achieving reliable and robust behaviour. In fact, any attempt to make the motion control more precise simply makes the robot more sensitive to any variations that do occur, as a result of environmental perturbations. Building a controller that can deal with the effects of such perturbations requires yet further complexity, again, none of which is actually of benefit in improving the performance of the robot—which was doing perfectly well in the first place.

We have now seen two examples in which using better technology make can the problem of harder. In the first case, reducing friction in the system removed important damping effects, which then had to be put back artificially, thus making the system more complicated and difficult to use. In the second case, increasing the resolution of the IR-sensors used, from one-bit of output to eight-bits of output, introduced more structural variation in signal profiles which, if it were to be reduced, would lead to not just a more complicated controller, but one which is likely to result in a less reliable and robust robot—since it would have to be more sensitive to such variations to reduce them.

4.1 Natural Sponges versus Natural Amplifiers

These are not two isolated examples. My claim is that this is how most of the real world works. Mechanical and electrical losses inevitable in systems built from physical media can act like sponges to soak up the variations they themselves induce in the behaviour of the system. If we construct systems from more accurate components so as to reduce losses, in an attempt to make them more energy efficient perhaps, or just to make them be 'better' systems, we can end up turning natural sponges into amplifiers of any intrinsic variations.

In this case we typically need to reintroduce the sponges somehow, which normally requires further system components, which in turn both increases the complexity of the system, and can make them harder to use successfully: as systems get more complicated they typically get harder to use. They can also become more sensitive to environmental conditions that were previously unproblematic, and often need more care in their calibration and recalibration. In general we can see that what are thought of as better systems can be worse systems as far as using them is concerned, or, more disturbing, can be less reliable and robust systems.

4.2 Sensing But Not Measuring

Are we then to conclude that increasing the accuracy of components or increasing the resolution of sensors we use on our robots is not a good idea? The answer is definately, no! Sensitivity and resolution are important in building robots. However, I want to suggest here that at least some of the difficulties experienced when using better robots are due to the approach taken to their design and use, and may thus be avoidable. In particular, I want to suggest that sensors on robots are not best understood as measuring devices, though this is how they are normally thought of. Conventionally, sensors are supposed to measure certain aspects of the environment of the robot, the distance to objects, for example: they are supposed to supply information about the state of the environment.

We can see, from the IR-sensor data presented in the previous section, that it would not be suitable to use in an attempt to acurately determine the location and direction of the robot at any particular time. There is so much structural variation in the IR sensors signals that to use them to decide what state the robot is in at any time would clearly be quite unreliable. Much too unreliable to successfully use them as measurements in some reinforcement learning scheme, or other world model-based navigation scheme, for example.

More generally, unless we introduce much better motion control into our robot, better motion control than is necessary for reliable and robust operation, and so reduce the structural variation in the sensor signals, we cannot take particular sensor signals to reliably and robustly correspond to particular robot-in-the-environment states: they cannot be taken as encoding information about the world, as is usually done. If we can't do this we can't define the reliable mappings from sensor signals to representations of robot-in-the-environment state. And, if we can't do this, we can't treat the sensor signals as measurements of anything: they simply vary in some way that depends upon the dynamics of the robot-environment interaction. Brooks, [Brooks, 91], describes similar experiences with real robots and, although he does not explicitly suggest sensors should not be treated as measuring devices, he does question the use and need of representations of the world, which require measurments in order to be built and maintained. See also, [Flynn & Brooks, 89], for further discussion and illustrations of how real robots can be made to work without trying to make sensors act as measuring devices.

4.3 Sponges, Filters, and the Right Interaction Dynamics

If the IR-sensor signals presented in section 4 are not being used as measurements what are they being used for? They are certainly used to influence the motion of the robot!

I think that a better way to view the behaviour of the LvII in its enclosure, and of agents acting in the world generally, is to consider the sensor signals as one aspect of an interaction process that is setup and maintained between the robot and its environment, and which involves other components such as the motors, control program, and physical structure of the robot. Viewed in this way, the role of sensors is to act as filters (c.f. [Wehner, 87]) on the detectable temporal changes that occur in any agent-environment interaction process, whose output is used to drive internal dynamical processes within the agent—implemented in software in the case of the LvII—which have appropriate sponge like properties to soak up unimportant variations in the filtered changes that in turn are used to influence the state of motor devices, which also have sponge and filtering properties of their own. Putting a robot together, viewed in this way, thus becomes a matter of getting the dynamics of interaction right by carefully designing, implementing, and adjusting, the filters, sponges, and internal dynamical processes, built out of both physical media, and computation.

By taking advantage of the particular properties of the physical devices used to build robots, we can, if we select and combine them well, get a lot of this filter and sponge work done intrinsically—for free. In a similar way we can also take advantage of the natural dynamics that any physical system has in getting, and keeping, the right interaction dynamics between our robot and its environment. Seen in this way, adaptive behvaiour thus becomes a problem of adjusting the filter and sponge propertise of the agent so as to maintain an effective agent-environment interaction dynamics. A good example of what, I think, can be understood as a good combination of filters and sponges in a robot control system, is Horswill's vision guided mobile robot, Poly, [Horswill, 93], which can demonstrate some relatively reliable and robust behaviour in a real world environment subject to significant variations, and which can present the robot with some unpredictable events and conditions[7].

I don't think Horswill's robot is best understood as an information processing system which uses sensors to measure aspects of its environment, and then decides what to do on the basis of this information. Treating agent-environments systems as dynamical system does not require this, see [Smithers, 94]. Mark Bickhard's interactivism model of emergent representation, [Bickhard, 93], presents an essentially similar and more thorough argument. Designing and building agent-environment systems as dynamical interaction systems which must maintain certain interaction dynamics is, I therefore suggest, a more appropriate approach to designing and building better robots, and one that does not introduce unnecessary complexity.

Acknowledgements

The work reported here was mostly carried out while the author held the SWIFT AI Chair at the VUB AI Lab, Brussels. Miles Pebody, Ian Porter, Piet Ruyssinck, and Danny Vereertbrugghen, were responsible for much of the development of the second generation Lego vehicle technology used. Anne Sjoström assisted in building, testing, and debugging the robot program used, and in carrying out the experiments from which the data were taken. Amaia Bernaras read and usefully commented on an earlier version of this paper. My thanks to all of them, and to two anonymous referees for their comments on the submitted draft of this paper. I am also happy to acknowledge the financial support of the University of the Basque Country for my current position, and the Faculty of Informatics, in particular, for providing my current academic home.

Notes

1. These are typically not simulations at all, not in the proper sense of the term, since they have never been validated against the robot and environment supposedly simulated. Indeed the robots and environments, mostly do not even exist, thus making any such validation impossible. What are widely referred to in the Artificial Intelligence, Artificial Life, animats, and robotics literature, as simulations, are better understood and described as computational models—often models of nonexistant robots and environments, but this is not a problem for modelling, only for simulation, which is a special type of modelling.

2. The fly-ball or centrifugal governor is so called because it uses two arms each with heavy balls attached at one end and pivoted at the other on a vertical rotating shaft. As the shaft speed increases, the balls fly out, and thus the arms swing up. As the shaft speed decreases, the balls fall in towards the shaft, and thus the arms swing down. This up and down movement of the arms is then used to open and close a value that controls the amount of steam supplied to the cylinder of a steam engine in such a way that a constant shaft speed is maintained.

3. Note that the fly-ball governor does not maintain the same speed of the main shaft as machines are brought into and out of use. It simply smooths the changes and reduces them to a minimum, given a maximum power output of the steam engine. This is because the relationship between the fly-ball governor and the throttle valve is fixed, so when a new machine is brought into use (and so draws power) the governor cannot return to exactly it's original speed since this would mean the value would be back to its original position and the amount of steam being let through would be as it was before, which is not enough to maintain the original speed under the additional load. With a powerfull enough steam engine and a well set governor and throttle valve relationship, this difference can be kept small and within acceptable limits.

4. Note that proportional integrating derivative (PID) controllers are also able to deal with the 'steady-state' problem: the failure of the Watt fly-ball governor to maintain the same main shaft speed, only a speed near the nominal speed. This *integrating* part of the controller acts against the *derivative* part used to counteract the hunting, or 'overshoot' problem. Consequently in any application of a PID controller a compromise has to be found between the setting of the parameters for each of these opposing aspects, and this can often be difficult to do in practice.

5. Second generation Lego vehicles are essentially the same
as first generation, see [Donnett & Smithers, 91], except that a two microporcessors architecture is used, one (a MC68HC11) to service the sensor and motor-control channels, [Vereertbrugghen, 93], and the second (a MC68340 with 0.5MByte RAM) to run the control program, together with 8 bit IR and light sensors (instead of 1 bit sensors), and a more efficient and flexible time-slicing runtime kernel, plus a number of other advanced features.

6. This uses a Hall-effect switch and six magnets placed in the front wheel with even spacing and alternating pole directions to produce three counts per complete revolution in one direction—changes in direction are detected in the program so that forward and reverse counts are maintained separately. The wheel diameter is 43mm, giving a circumference of 135.089mm, and thus a distance per wheel count of 45.03mm. Having the front wheel fixed (i.e., not a caster) means that it is always in line with the direction of motion or tangential to it. This means that, using this wheel counter, it is possible to reliably estimate the distance travelled: essentially it integrates only the forward motion of the robot, much as a planimeter does—an instrument used for measuring the area of closed plane figures, such as ship hull sections, etc.

7. Though Poly can't deal with everything that can happen. Ian was kind enough to give me a demonstration of his robot when I visited him. It happened to be one of the first warn sunny days of the year in Cambridge, so people were leaving their office doors open. This meant there were light patches on the floor which had not been there before–when Poly was being developed and tested–and these sometimes caused the

robot to get confused about where it was. The situation was not helped any by the fact that over night all the overhead lighting had been changed (for more efficient tubes) causing thr ambient light level to be a little different from before, and so affecting some of the image processing involved—a good illustration of the kinds of thing real robots have to be able to cope with.

References

[Bickhard, 93] Mark H. Bickhard, 1994. Representation Content in Humans and Machines, Journal of Experimental and Theoretical AI, vol 5, pp 285–333.

[Brooks, 91] Rodney A. Brooks, 1991. Intelligence Without Representation, Artificial Intelligence, Vol 47, no 1–3, special issue, pp 139–159.

[Dickinson & Jenkins, 91] H. W. Dickinson and Rhys Jenkins, 1981. *James Watt and the Steam Engine*, Encore Editions, London. First published 1927.

[Donnett & Smithers, 91] Jim Donnett and Tim Smithers, 1991. A Technology for Studying Inteligent Systems, in Jean-Arcady Meyer and Stewart W. Wilson (eds.), *From Animals to Animats*, proceedings of the First International Conference on Simulation of Adaptive Behaviour, The MIT Press, Cambridge, Mass., pp 540–549.

[Flynn & Brooks, 89] Anita M. Flynn and Rodney A. Brooks, 1989. Battling Reality, AI Memo 1148, MIT AI Labrotory, October 1989.

[Horswill, 93] Ian Horswill, PhD thesis, MIT AI Laboratory, 1993.

[Kurzweil, 90] Raymond Kurzweil, 1990. *The Age of Intelligent Machines*, The MIT Press, Cambridge, Mass.

[Maxwell, 68] James Clerk Maxwell, 1868. On Governors, *Proceedings of the Royal Society*, 16 (1867/68), pp 270–283. Reprinted in Richard Bellman and Robert Kalaba (eds.), *Mathematical Trends in Control Theory*, New York, 1964, pp 3-17.

[Mayr, 70] Otto Mayr, 1970. *The Origins of Feedback Control*, The MIT Press, Cambridge, Mass.

[Nehmzow & Smithers, 91] Ulrich Nehmzow and Tim Smithers, 1991, Mapbuilding using Self-Organising Networks in "Really Useful Robots", in Jean-Arcady Meyer and Stewart W. Wilson (eds.), *From Animals to Animats*, proceedings of the First International Conference on Simulation of Adaptive Behaviour, The MIT Press, Cambridge, Mass., pp 152–159.

[Nehmzow & Smithers, 92] Ulrich Nehmzow and Tim Smithers, 1992. Using Motor Actions for Location Recognition, in Francisco J Varela and Paul Bourgine (eds.), *Towards a Practice of Autonomous Systems*, proceedings of the First European Conference on Artificial Life, The MIT Press, Cambridge, Mass., pp 96–104.

[Smithers, 92] Tim Smithers, 1992. Taking Eliminative Materialism Seriously: A Methodology for Autonomous Systems Research, in Francisco J Varela and Paul Bourgine (eds.), *Towards a Practice of Autonomous Systems*, proceedings of the First European Conference on Artificial Life, The MIT Press, Cambridge, Mass., pp 31–40.

[Smithers, 93] Tim Smithers, 1993. On Behaviour as Dissipative Structures in Agent-Environment Interactions Processes, presented at the workshop *prerational Intelligence: Phenomenology of Complexity Emerging in Systems of Agents Interacting Using Simple Rules*, held at the Centre for Interdisciplinary Studies (ZiF), Bielefeld, November 22–26, 1993, as part of the Prerational intelligence Research Project.

[Smithers, 94] Tim Smithers, 1994. On Agent-Environment Systems, paper submitted to Artificial Life IV, MIT, Cambridge, July, 1994.

[Vereertbrugghen, 93] Dany Vereertbrugghen, 1993. *Design and Implementation of a Sensor-Motor Control Unit for Mobile Robots*, Licentie Thesis, AI Laboratory, Vrije Universiteit Brussel.

[Wehner, 87] Rüdiger Wehner, 1987. Matched Filters—Neural Models of the External World, *Journal Compputational Physiology, A*, vol 161, pp 511–531.