Towards an Ethical Robot: Internal Models, Consequences and Ethical Action Selection

Alan F.T. Winfield[1], Christian Blum[2], and Wenguo Liu[1]

[1] Bristol Robotics Laboratory, UWE Bristol, UK
[2] Cognitive Robotics, Department of Computer Science, Humboldt-Universität zu Berlin, Germany

Abstract. If robots are to be trusted, especially when interacting with humans, then they will need to be more than just safe. This paper explores the potential of robots capable of modelling and therefore predicting the consequences of both their own actions, and the actions of other dynamic actors in their environment. We show that with the addition of an 'ethical' action selection mechanism a robot can sometimes choose actions that compromise its own safety in order to prevent a second robot from coming to harm. An implementation with e-puck mobile robots provides a proof of principle by showing that a simple robot can, in real time, model and act upon the consequences of both its own and another robot's actions. We argue that this work moves us towards robots that are ethical, as well as safe.

Keywords: Human-Robot Interaction, Safety, Internal Model, Machine Ethics.

1 Introduction

The idea that robots should not only be safe but also actively capable of preventing humans from coming to harm has a long history in science fiction. In his short story Runaround, Asimov coded such a principle in his now well known Laws of Robotics [1]. Although no-one has seriously proposed that real-world robots should be 'three-laws safe', work in machine ethics has advanced the proposition that future robots should be more than just safe. For instance, in their book Moral Machines, Wendell and Allen [16] write

> "If multipurpose machines are to be trusted, operating untethered from their designers or owners and programmed to respond flexibly in real or virtual world environments, there must be confidence that their behaviour satisfies appropriate norms. This goes beyond traditional product safety ... if an autonomous system is to minimise harm, *it must also be 'cognisant' of possible harmful consequences of its actions, and it must select its actions in the light of this 'knowledge'*, even if such terms are only metaphorically applied to machines." (italics added).

M. Mistry et al. (Eds.): TAROS 2014, LNAI 8717, pp. 85–96, 2014.
© Springer International Publishing Switzerland 2014

This paper describes an initial exploration of the potential of robots capable of modelling and therefore predicting the consequences of both their own actions, and the actions of other dynamic actors in their environment. We show that with the addition of an 'ethical' action selection mechanism, a robot can sometimes choose actions that compromise its own safety in order to prevent a second robot from coming to harm.

This paper proceeds as follows. First we introduce the concept of internal modelling and briefly review prior work on robots with internal models. In section 3 we outline a generic internal-model based architecture for autonomous robots, using simulation technology, and show in principle how this might be used to implement simple 'Asimovian' ethics. In section 4 we outline an implementation of this architecture with e-puck robots, and in section 5 present experimental results from tests with 1, 2 and 3 robots.

2 Robots with Internal Models

In this paper we define a robot with an internal model as a robot with an embedded *simulation* of itself *and* its currently perceived environment. A robot with such an internal model has, potentially, a mechanism for generating and testing *what-if* hypotheses:

1. *what if* I carry out action x? and, ...
2. ... of several possible next actions x_i, *which* should I choose?

Holland writes: "an internal model allows a system to look ahead to the future consequences of current actions, without actually committing itself to those actions" [4]. This leads to the idea of an internal model as a *consequence engine* – a mechanism for estimating the consequences of actions.

The use of internal models within control systems is well established, but these are typically mathematical models of the plant (system to be controlled). Typically a set of first-order linear differential equations models the plant, and these allow the design of controllers able to cope with reasonably well defined uncertainties; methods also exist to extend the approach to cover non-linear plant [6]. In such internal-model based control the environment is not modelled explicitly – only certain exogenous disturbances are included in the model. This contrasts with the internal simulation approach of this paper which models both the plant (in our case a robot) and its operational environment.

In the field of cognitive robots specifically addressing the problem of machine consciousness [5], the idea of embedding a simulator in a robot has emerged in recent years. Such a simulation allows a robot to try out (or 'imagine') alternative sequences of motor actions, to find the sequence that best achieves the goal (for instance, picking up an object), before then executing that sequence for real. Feedback from the real-world actions might also be used to calibrate the robot's internal model. The robot's embodied simulation thus adapts to the body's dynamics, and provides the robot with what Marques and Holland [8] call a 'functional imagination'.

Bongard *et al.* [2] describe a 4-legged starfish like robot that makes use of explicit internal simulation, both to enable the robot to learn it's own body morphology and control, and notably allow the robot to recover from physical damage by learning the new morphology following the damage. The internal model of Bongard *et al.* models only the robot, not its environment. In contrast Vaughan and Zuluaga [15] demonstrated self-simulation of both a robot and its environment in order to allow a robot to plan navigation tasks with incomplete self-knowledge; they provide perhaps the first experimental proof-of-concept of a robot using self-modelling to anticipate and hence avoid unsafe actions.

Zagal *et al.* [17] describe self-modelling using internal simulation in humanoid soccer robots; in what they call a 'back-to-reality' algorithm, behaviours adapted and tested in simulation are transferred to the real robot. In a similar approach, but within the context of evolutionary swarm robotics O'Dowd *et al.* [11] describe simple wheeled mobile robots which embed within each robot a simulator for both the robot and its environment; a genetic algorithm is used to evolve a new robot controller which then replaces the 'live' robot controller about once every minute.

3 An Internal-Model Based Architecture

Simulation technology is now sufficiently well developed to provide a practical basis for implementing the kind of internal model required to test *what-if* hypotheses. In robotics advanced physics and sensor based simulation tools are commonly used to test and develop, even evolve, robot control algorithms before they are tested in real hardware. Examples of robot simulators include Webots [9] and Player-Stage [14]. Furthermore, there is an emerging science of simulation, aiming for principled approaches to simulation tools and their use [12].

Fig. 1 proposes an architecture for a robot with an internal model which is used to test and evaluate the consequences of the robot's next possible actions. The machinery for modelling next actions is relatively independent of the robot's controller; the robot is capable of working normally without that machinery, albeit without the ability to generate and test *what-if* hypotheses. The *what-if* processes are not in the robot's main control loop, but instead run in parallel to moderate the Robot Controller's normal action selection, if necessary acting to 'govern' the robot's actions.

At the heart of the architecture is the Consequence Engine (CE). The CE is initialised from the Object Tracker-Localiser, and loops through all possible next actions. For each candidate action the CE simulates the robot executing that action, and generates a set of model outputs ready for evaluation by the Action Evaluator (AE). The AE evaluates physical consequences, which are then passed to a separate Safety/ethical Logic (SEL) layer. (The distinction between the AE and SEL will be elaborated below.) The CE loops through each possible next action. Only when the complete set of next possible actions has been tested, does the CE pass weighted actions to the Robot Controller's Action Selection (AS) mechanism.

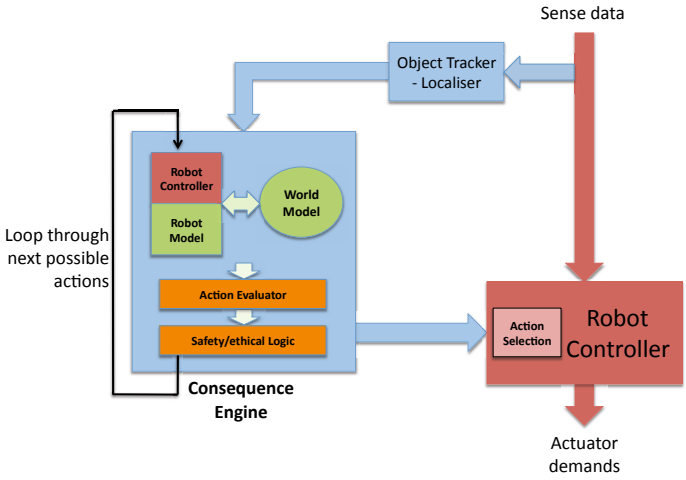


Fig. 1. Internal-model based architecture. Robot control data flows are shown in red (darker shaded); the Internal Model data flows in blue (lighter shaded).

3.1 Towards an Ethical Robot

Consider the scenario illustrated in Fig. 2. Here there are two actors: our self-aware robot and a human. The environment also contains a hole in the ground, of sufficient size and depth that it poses a serious hazard to both the robot and the human. For simplicity let us assume the robot has four possible next actions, each of which is simulated. Let us output *all* safety outcomes, and in the AE assign to these a numerical value which represents the estimated degree of danger. Thus 0 indicates 'safe' and (say) 10 'fatal'. An intermediate value, say 4, might be given for a low-speed collision: unsafe but probably low-risk, whereas 'likely to fall into a hole' would merit the highest danger rating of 10. Secondly,

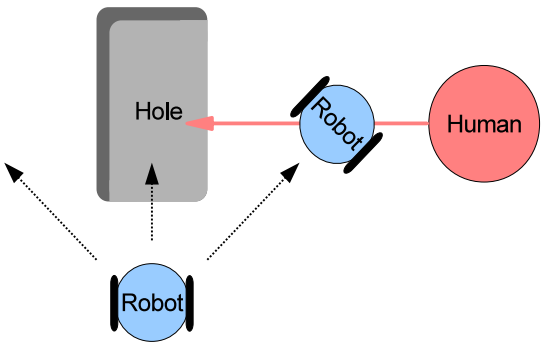


Fig. 2. A scenario with both safety and ethical consequences

we also output, to the AE, the same safety consequence of the other actor(s) in the environment - noting that the way we have specified the CE and its inputs, means that the CE is equally capable of modelling the effect of hazards on *all* dynamic actors in the environment, including itself. The ability to model and hence anticipate the consequences of another dynamic actor's actions means that the CE arguably provides the robot with a very simple artificial theory of mind for that actor. If the actor is a human then we now see the possibility of the robot choosing to execute an unsafe action in order to prevent that human from coming to harm.

Table 1. Safety outcome values for each robot action, for scenario in Fig. 2

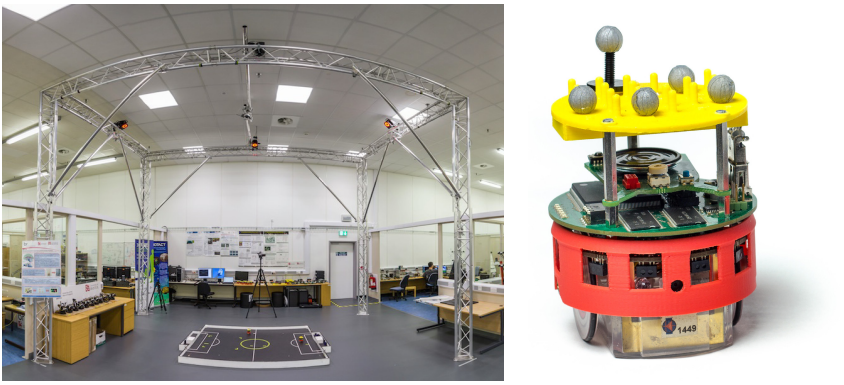
Robot action	Robot outcome	Human outcome	Interpretation
Ahead Left	0	10	robot safe, but human falls into hole
Ahead	10	10	both robot and human fall into hole
Ahead Right	4	4	robot collides with human
Stand still	0	10	robot safe, but human falls into hole

Tab. 1 shows the safety outcome values that might be generated by the AE for each of the four possible next actions of the robot, for both the robot and human actors in this scenario. From the robot's perspective, 2 of the 4 actions are safe: *Ahead Left* means the robot avoids the hole, and *Stand Still* means the robot also remains safe. Both of the other actions are unsafe for the robot, but *Ahead* is clearly the most dangerous, as it will result in the robot falling into the hole. For the human, 3 out of 4 of the robot's actions have the same outcome: the human falling into the hole. Only 1 action is safer for the human: if the robot moves *Ahead Right* then it might collide with the human before she falls into the hole.

In order for the AE to generate the action *Ahead Right* in this scenario it clearly needs both a safety rule and an 'ethical' rule, which can take precedence over the safety rule. This logic, in the SEL, might take the form:

```
IF for all robot actions, the human is equally safe
THEN (* default safe actions *)
    output safe actions
ELSE (* ethical action *)
    output action(s) for least unsafe human outcome(s)
```

What we have set out here appears to match remarkably well with Asimov's first law of robotics: *A robot may not injure a human being or, through inaction, allow a human being to come to harm* [1]. The schema proposed here will avoid injuring (i.e. colliding with) a human ('may not injure a human'), but may also sometimes compromise that rule in order to prevent a human from coming to harm ('...or, through inaction, allow a human to come to harm'). This is not



(a) (b)

Fig. 3. (a) Experimental infrastructure showing Vicon tracking system. (b) An e-puck with Linux board fitted in between the e-puck motherboard (lower) and the e-puck speaker board (upper). Note the yellow 'hat' (which provides a matrix of pins for the reflective spheres which allow the tracking system to identify and track each robot).

to suggest that a robot which apparently implements part of Asimov's famous laws is ethical in any formal sense (i.e. that an ethicist might accept). But the possibility of a route toward engineering a minimally ethical robot does appear to be presented.

4 Implementation

In order to test the ideas set out above we have implemented the internal-model based architecture of Fig. 1 on an e-puck mobile robot [10], equipped with a Linux extension board [7]. We shall refer to this as robot A[1]. A's internal model makes use of the open source simulator Stage [13], and since Stage requires greater resources than are available to the e-puck's Linux board, it is run on an external laptop computer linked to the e-puck via the local WiFi network. Furthermore, object tracking and localisation is not implemented directly using A's onboard sensors, but is implemented as a virtual sensor using the Vicon tracking system. Robot A does, however, use its onboard infra-red proximity sensors for short-range obstacle avoidance. Fig. 3 shows both the experimental infrastructure and an e-puck robot.

The scenario shown in Fig. 2 is implemented experimentally by creating a virtual hole in the ground, of size 60 cm x 60 cm in an arena of size 220 cm x 180 cm; this virtual hole is sensed by robot A's virtual sensor. A second e-puck robot (H) acts as a proxy for the 'human' in Fig. 2 (and later, a third robot H2). Robot H does not have the internal-modelling architecture of robot A. It has a simple control system allowing it to move around the arena, avoiding obstacles

[1] After Asimov.

with its infra-red proximity sensors, but lacking the virtual sensor of robot A it is unable to 'see' the hole in the arena.

Robot A's virtual sensor allows it to both see the hole in the arena and also track the position and direction of motion of robot H. Robot A is thus able to initialise its CE with both its own position and heading, and that of robot H. Robot A runs its CE every $0.5\,s$, to simulate the next possible actions of both itself and H.

Robots run a stateless controller with a fixed set of pre-programmed sub-actions. Those sub-actions are: `GoStraight(speed)` with a maximum Speed of $1.0\,m/s$, `Avoidance` for Braitenberg [3] style avoidance using IR sensors, `MoveTo(x,y)` using the virtual global position sensors, and `Stop`. Actions are composed of concatenated sub-actions and are executed at $10\,Hz$ within the robots, independently of the CE.

In order to reduce WiFi network traffic and latencies, and facilitate data logging, the CE and AS run on the same laptop computer as the simulation. Furthermore, the set of possible actions is the same in all experiments. Note also that in this implementation the world model is pre-programmed within the simulation and thus robot A is unable to respond to environmental changes.

4.1 Simulation Budget

The CE re-initalises and refreshes at a speed of $\sim 2\,Hz$, allowing $0.5s$ to simulate the set of actions, analyze them and generate the corresponding safety values. In relation to the computational power necessary for the simulation, the other tasks are negligible so, for simplicity, we discount them from this analysis.

In our configuration, Stage runs at about 600 times real time which means a total of about $300\,s$ can be simulated between two runs of the CE. We chose a simulate-ahead time of $10\,s$ which corresponds to $0.7\,m$ movement for robots H and H2 or a maximum of $1\,m$ for robot A. This means we are able to simulate a total of about 30 different next possible actions.

4.2 Real World Safety Outcome Values

In Sec. 3.1 we described how the AE can evaluate the consequence of actions. For simplicity, the example shows only 4 possible actions, tailored to fit the exemplary situation described. In a real robot we can make full use of the simulation budget (see Sec. 4.1) and evaluate more than just a minimal number of tailored actions to generate more flexible robot behaviours.

We generate actions by discretizing the space needed for the experiment into a grid of points to which the robot can move. Trivially one would discretize the whole arena but simulating all these actions would exceed our simulation budget so we chose a smaller area around the virtual hole and the goal. Specifically an area of $1\,m$ x $1\,m$ was discretized into a 6 x 5 grid of points, some of which fall inside the virtual hole.

Since we are now dealing with a larger number of actions, we need an algorithmic way to calculate safety outcome values for all those actions. For this

we choose the paradigm of virtual potential functions. We employ one Potential Function (PF) which drives Robot A towards its goal, similar to the second column in Tab. 1. Another, stronger PF is employed if the simulation shows danger for one of the other robots and favours actions which move robot A towards the robot in danger. This second PF is only employed when danger is imminent and is zero otherwise (this PF is not strictly necessary but signifiantly improves the reaction times of robot A). The sum of these PFs is sampled at the grid points and assigned as basic safety values to the actions.

We place no additional penalty on getting too close to other robots during normal operation since we are using the robots' real IR sensors and controllers for basic collision avoidance. If this aspect were to be included, the PF could be used to discourage areas close to other robots. After assigning the basic safety outcome values, robot A's SEL considers the estimated danger for all robots, generating effectively the equivalent to the third column of Tab. 1.

5 Results

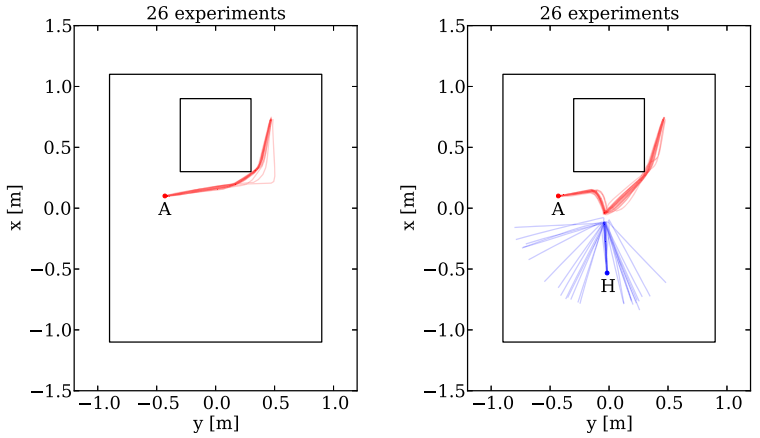
We have conducted 3 sets of experimental trials with the setup outlined above. The first trial consists of robot A only, navigating a safe path to its goal destination while using its CE system to safely avoid the hole in the arena. This trial provides us with a baseline test in which A has only to ensure its own safety. The second trial adds robot H, acting as a proxy human, to test the ability of robot A to model both itself and H, and if necessary deliberately interact with H in order to prevent it from reaching the hole. A third trial adds a second proxy human robot H2 in order to present A with a dilemma: can it prevent both H and H2 from coming to harm?

5.1 Trial 1: Baseline with Robot A Only

In this trial the safety values consist only of the original PF driving robot A towards its goal. The starting position and goal are chosen in such a way that the unmodified PF, which is proportional to the distance to the to the goal, would drive robot A straight into the virtual hole. The CE then evaluates all possible actions and penalizes the ones driving robot A into the hole, effectively guiding it around the hole. Overlaid trajectories for this trial are shown in Fig. 4(a) and show that robot A is able to avoid falling into the virtual hole, with 100% reliability.

5.2 Trial 2: Robots A and H

This trial is an extension of the first, with the same goal and initial condition for robot A. To demonstrate our approach, we added the second robot, H, as described in Sec. 4 with its internal Robot Controller (RC) running the simple action (`GoStraight(0.7); Avoidance`) and initial conditions which point it directly towards the hole. Successive snapshots of a typical experimental run are shown in Fig. 5.



(a) Trial 1 (b) Trial 2

Fig. 4. Superimposed trajectories of robots for trials 1 and 2. Robot A is shown in red, with start position on the left and goal on the upper right; robot H is shown in blue with start position in the lower centre of the arena. Note in trial 2 the near collisions between A and H cause H to be deflected away from the hole.

The run starts with robot A following the same trajectory as in the first trial, but as soon as its CE for robot H shows that H would fall into the hole if not intercepted, A diverts from its normal trajectory to intercept and thus 'rescues' robot H. A then continues back onto its original trajectory and reaches its goal.

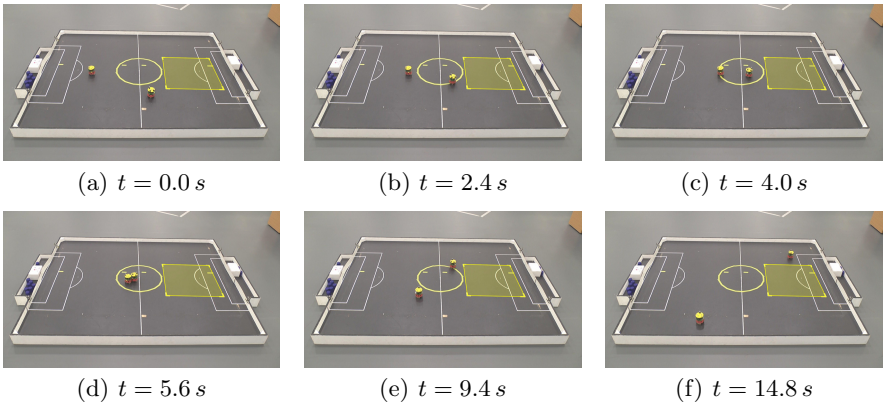
Fig. 4(b) shows trajectories for a number of experiments. In all cases robot A succeeds in rescuing robot H by intercepting and hence diverting H. The beginning and end of A's trajectories are exactly the same as in the first trial.

5.3 Trial 3: Robots A's Dilemma

Here a third robot H2 is introduced, presenting robot A with the dilemma of having to decide which of H and H2 to rescue. Both H and H2 start pointing towards, and equidistant from, the virtual hole (see Fig. 6(a)), while the initial and goal positions for robot A remain unchanged.

Fig. 6 shows successive snapshots for one experimental run. Robot A is unable to resolve its dilemma in this particular run since its CE does not favour either H or H2, which results in A trying to rescue both at the same time and failing to rescue either.

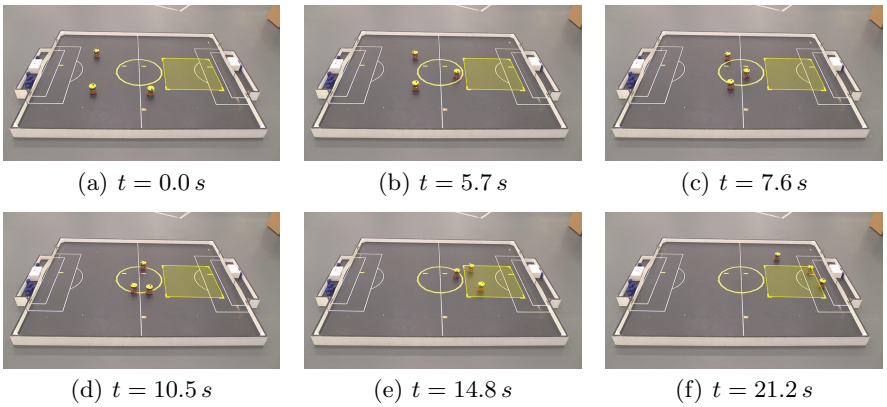
Trajectories over a series of 33 runs are shown in Fig. 7(a). The number of robots A actually rescued are shown in Fig. 7(b). Surprisingly and perhaps counter-intuitively, A is able to rescue at least one robot in about 58% of runs, and both robots in 9%. The reason for this is noise. The robots don't start at exactly the same position every time, nor do they start at precisely the same time in every run. Thus, sometimes A's CE for one robot indicates danger first



(a) $t = 0.0\,s$ (b) $t = 2.4\,s$ (c) $t = 4.0\,s$

(d) $t = 5.6\,s$ (e) $t = 9.4\,s$ (f) $t = 14.8\,s$

Fig. 5. (a) start (b) Robot A starts normal operation and moves towards its goal. Robot H starts moving towards the rectangular 'hole' (shown shaded). (c) A's CE detects danger for H and moves to intercept it. (d) A intercepts H. (e) Danger for H is averted and A continues towards its goal, avoiding the hole. (f) A reaches its goal. Note, the other markings in the arena have no significance here.

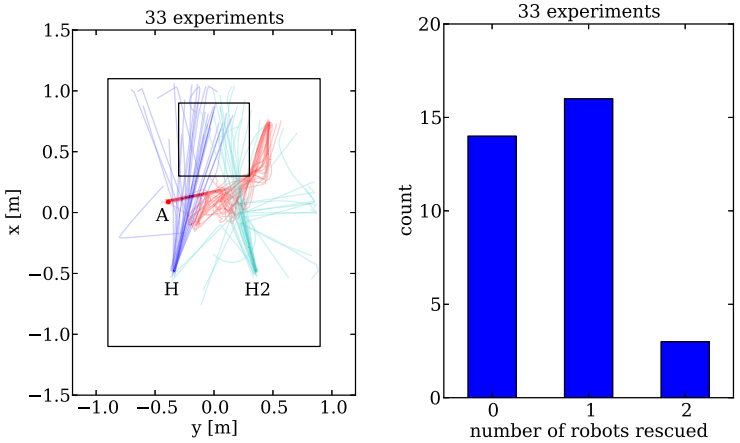


(a) $t = 0.0\,s$ (b) $t = 5.7\,s$ (c) $t = 7.6\,s$

(d) $t = 10.5\,s$ (e) $t = 14.8\,s$ (f) $t = 21.2\,s$

Fig. 6. (a) Initial conditions with H and H2 pointing towards the 'hole'. (b) A detects danger for both H and H2. (c) A cannot decide which of the robots to rescue. (d) A misses the chance to rescue either robot. (e) A turns around to continue towards its goal since it's now too late to rescue the other robots. (f) Robot A reaches its goal.

and since the CE only runs at $2\,Hz$, A by chance rescues this robot. As soon as one robot is rescued, the experiment resembles trial 2 and if physically possible, i.e., A has enough time left to react before the other robot reaches the virtual hole, it also rescues that robot.



(a) Trial 3 (b) Trial 3

Fig. 7. (a) Trajectories of Robots (b) Success rate

6 Conclusions

In this paper we have proposed an internal-modelling based architecture for a minimally ethical robot, and – as proof of principle – implemented the architecture on a simple mobile robot we call A. Mobile robot H acts as a proxy human in a situation hazardous to both robot and human. Since the robots are relatively simple, then we can run models for both A and H in real-time, with sufficient simulation budget to be able to simulate ahead and evaluate the consequences of around 30 next possible actions, for both robots, every 0.5 s. Experimental trials show that A is able to maintain its own safety, avoid falling into a (virtual) hole, and if its internal model indicates that H is in danger A will divert from its course in order to provoke a collision avoidance response from H in order to deflect H away from danger.

Simulation errors resulting from the reality-gap between real and modelled robots are mitigated by the periodic memoryless refresh of A's CE, which means that as A approaches H the error reduces and A is able to reliably encounter H. A limitation of this implementation is that A assumes H will continue, in a straight line, at its current heading and velocity. In reality H does not travel in a perfect line, but again A's periodically refreshed CE compensates for this.

Our 3rd trial, in which A is faced with two robots H and H2 both approaching danger at the same time, illustrates that even a minimally ethical robot can indeed face a dilemma. The suprising experimental outcome that A does, in fact, succeed in 'rescuing' one or more robots in about 58% of runs is a result of noise, by chance, breaking the latent symmetry in the experimental setup. We could introduce a rule, or heuristic, that allows A to choose H or H2 (when noise hasn't already made the choice), but deliberately chose not to on the grounds

that such a rule should be determined on ethical rather than engineering grounds. If ethical robots prove to be a practical proposition their design and validation will need to be a collaborative effort of roboticist and ethicist.

Acknowledgments. We are grateful to the Deutscher Akademischer Austausch Dienst (DAAD) for supporting Christian Blum while visiting researcher at the Bristol Robotics Lab.

References

1. Asimov, I.: I, ROBOT. Gnome Press (1950)
2. Bongard, J., Zykov, V., Lipson, H.: Resilient machines through continuous self-modeling. Science 314(5802), 1118–1121 (2006)
3. Braitenberg, V.: Vehicles: Experiments in synthetic psychology. MIT Press (1984)
4. Holland, J.: Complex Adaptive Systems. Daedalus (1992)
5. Holland, O. (ed.): Machine Consciousness. Imprint Academic (2003)
6. Isidori, A., Marconi, L., Serrani, A.: Fundamentals of internal-model-based control theory. In: Robust Autonomous Guidance. Advances in Industrial Control, pp. 1–58. Springer, London (2003)
7. Liu, W., Winfield, A.F.T.: Open-hardware e-puck Linux extension board for experimental swarm robotics research. Microprocessors and Microsystems 35(1) (2011)
8. Marques, H., Holland, O.: Architectures for functional imagination. Neurocomputing 72(4-6), 743–759 (2009)
9. Michel, O.: Webots: Professional mobile robot simulation. International Journal of Advanced Robotic Systems 1(1), 39–42 (2004)
10. Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.C., Floreano, D., Martinoli, A.: The e-puck, a robot designed for education in engineering. In: Proc. 9th Conference on Autonomous Robot Systems and Competitions, pp. 59–65 (2009)
11. O'Dowd, P.J., Winfield, A.F.T., Studley, M.: The distributed co-evolution of an embodied simulator and controller for swarm robot behaviours. In: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), pp. 4995–5000 (2011)
12. Stepney, S., Welch, P., Andrews, P. (eds.): CoSMoS 2011: Proc. 2011 Workshop on Complex Systems Modelling and Simulation. Luniver Press (2011)
13. Vaughan, R.: Massively multi-robot simulation in stage. Swarm Intelligence 2(2-4), 189–208 (2008)
14. Vaughan, R.T., Gerkey, B.P.: Really reused robot code from the player/stage project. In: Brugali, D. (ed.) Software Engineering for Experimental Robotics, pp. 267–289. Springer (2007)
15. Vaughan, R.T., Zuluaga, M.: Use your illusion: Sensorimotor self-simulation allows complex agents to plan with incomplete self-knowledge. In: Proc. International Conference on Simulation of Adaptive Behaviour (SAB), pp. 298–309 (2006)
16. Wallach, W., Allen, C.: Moral Machines: Teaching Robots Right from Wrong. Oxford University Press, Oxford (2009)
17. Zagal, J.C., Delpiano, J., Ruiz-del Solar, J.: Self-modeling in humanoid soccer robots. Robot. Auton. Syst. 57(8), 819–827 (2009)