

Autonomy Without Independence: Animal Training as a Model for Robot Design

David C. Wyland

Reasonable Machines
165 Berkshire Drive
Morgan Hill, CA 95037 USA
dcwyland@ix.netcom.com

Abstract. A classic autonomous robot is an autonomous agent for open, unpredictable environments. Such an agent is inherently autonomous but not independent. Independence implies unpredictability, which is incompatible with agency. The current robot models – behavior based and artificial intelligence – have not been effective at implementing the classic autonomous robot model due to limitations in their definitions. The artificial intelligence model cannot deal with unpredictable environments, and neither model directly includes the concept of agency. Animal training as a model for robotics has the potential to avoid these problems. Animal training has several advantages. It has an inherent model of agency. Goals and behavior are formally separated into human goals and animal behavior. The animal is autonomous, requiring conversation between human and animal, but it is not an independent entity. A robot designed using this model is an articulate machine, programmed as an agent for the user.

1 Introduction

Animals can be trained to do amazing things. Examples abound in the form of the sheep dog, the cutting horse, the rescue dog, the working elephant and the seeing-eye dog, to say nothing of circus animals. Animals have been trained for various complex human support tasks for millennia.

The purpose of animal training, such as dog training, is to cause the animal to exhibit behaviors on command that will achieve the goals of the trainer. The trainer does this by enabling and modifying existing behaviors to create new ones.

When properly trained, a sheep dog will reliably do the sheep herding tasks given by its master, the shepherd. While working, the sheep dog acts as an agent for the shepherd, not as an independent animal. If a rabbit runs across the sheep dog's path, the dog continues

herding instead of chasing the rabbit. This model of interaction between human and animal has some interesting characteristics.

1. The dog is trained as an agent for the human. The human has the goals in the relationship; the dog does not.
2. The human does the task design, selecting and modifying behaviors to achieve the desired goals.
3. The dog is not required to be intelligent nor conscious to be useful to us.
4. Tasks and their goals are formally separated from behavior.
5. The dog works in an open environment.
6. The behavior of the dog is probabilistic. It may not achieve a task on first try, and it will achieve it to a degree that can vary from one time to the next.

Let us consider this as a model for a robot, where the robot takes the place of the dog. As in the above examples, the human is the trainer. We will assume a Behavior Based (BB) robot with the added ability to select and modify the behavior in real time by communication with the robot. If we implement the model correctly, it will have the following characteristics:

1. The robot is designed as an agent for the human. The human has the goals; the robot does not.
2. The human does the task design, selecting and modifying behaviors to achieve the desired goals.
3. The robot is not required to be intelligent nor conscious.
4. Tasks and their goals are formally separated from behavior.
5. Because it is behavior based, the robot works in an open environment.
6. The behavior of the robot is probabilistic. Because it works in an open, unpredictable environment, it may not achieve a task on first try, and it will achieve it to a degree that can vary from one time to the next.

The Animal Training (AT) model provides autonomy without independence. The robot acts autonomously in open, unpredictable environments, but its actions are completely dependent upon human design and direction. Rather than having an independent intelligence, we have a perfect dependent agent, one with no drives other than those we define and enable. This paper compares the AT model for robot design against previous design models in terms of their usefulness for designing a classic autonomous robot.

1.1 The Robot as an Agent for Open Environments

We can define an autonomous mobile robot as an agent for open environments, as discussed by Wyland in [1]. The robot as an agent for humans is inherent in our concept of the robot. We want it to do work for us. An open environment is unpredictable but familiar, and knowledge of the environment is inherently and chronically incomplete. It is unpredictable because events can happen spontaneously, such as a human or moving object interacting with the robot. The robot must operate on the basis of familiarity

because it is never in exactly the same place twice and it never perceives its environment in exactly the same way twice. A robot working in an unpredictable environment implies autonomy. It must deal with its environment as it perceives it.

1.2 Agency Versus Independence

To an observer, an autonomous robot appears to “think for itself” as an independent entity as it reacts to its unpredictable environment. However, if we know its internal programming well enough to accurately predict how it will react to its environment, we no longer consider the robot as thinking for itself, but consider it as a predictable automatic machine. Therefore, a robot or other entity (animal, human, etc.) is independent to the degree that we cannot accurately predict how it will react to its environment.

We want the robot to be predictable in all environments. Unpredictable behavior in any environment could be dangerous. Properly designed, the robot will act in a predictable manner for a given task in a familiar environment. We also want it to act in a predictable manner in an unfamiliar environment. If the robot is in an unfamiliar environment, we want it to have a well-defined, default behavior, such as stopping and asking for help.

Robots can be unpredictable in unfamiliar environments if not designed for them. To be predictable in an unfamiliar environment, the robot must detect when the current environment is sufficiently different from the environments it knows and respond with a default behavior. Many machine-learning mechanisms, such as neural networks and genetic algorithms, do not inherently have such a novelty detection mechanism and are vulnerable to unpredictable behavior. Even simple task and behavior designs need to verify that the current environment is suitable for current task to remain predictable.

The concepts of agency and independence are incompatible. Agency means that the agent works for the user, and the device makes predictable decisions defined by the user. Independence means that the independent device makes unpredictable decisions on its own. An entity can be an agent to the degree that it is predictable; it is independent to the degree that it is unpredictable. We can make an independent entity an agent only to the degree that we make it predictable, i.e. to the degree that we eliminate its independence.

An agent needs to be autonomous but not independent. Autonomy and independence are different concepts. Independence implies autonomy, but autonomy does not imply independence. To be independent, you must be autonomous, or the concept has little meaning. However, you can have autonomy without independence. A thermostat is an example of autonomy without independence. It autonomously decides when to turn the heater on and off, but its user sets its temperature goal. It serves as an agent for the user. A sheep dog has autonomy but not independence. It does the tasks the shepherd assigns.

We want our robots to be autonomous but dependent machines because we want them to be safe and reliable. This means that for a given environment and task, the robot’s actions are completely predictable.

3 Robot Design Models

A design model captures the problem to be solved by the design and the approach to solving it. To select a model, we need to identify the requirements of the design. In our discussion of robots, we defined them as agents for open environments.

The first requirement is to be able to work in an open environment. An open environment is unpredictable but familiar. Homes, offices and the outdoors are examples of open environments. These are open because people and animals come and go, and furniture is rearranged. They are also open because the robot will never be in exactly the same place twice and will never see its environment from exactly the same pose twice.

The second requirement is agency. Agency is the ability to carry out useful tasks for others, e.g. humans. As we require this ability in a robot, we need to be specific about the qualities of agency we require. We want to be able to give the robot a task at any time and have it respond in a timely manner, where timely is measured against human performance. If adding a new task requires days or weeks of design and implementation time, the robot does not meet our requirement of timely response.

3.1 Current Design Models

There have been two major models in robot design: the artificial intelligence (AI) model and the Behavior-Based (BB) model. The AI model is concerned with modeling human intelligence, as its name implies. The image of the completed AI model is an android, a synthetic human. The BB model is derived from the cybernetic tradition as defined by Weiner in [2], which is concerned with communication and control in human and animal behavior. Its image is a synthetic animal or insect.

3.2 Artificial Intelligence (AI) Model

Artificial Intelligence (AI) as a formal discipline developed a particular model for robot design called the Sense-Model-Plan-Act, or SMPA model, as discussed by Brooks in [3]. You sense (i.e., measure) the environment, incorporate the measurements into a model of your world, create a plan to execute a given task by analyzing that model, and carry out the actions defined by the plan. If nothing in the environment changes, you go through this sequence once to execute a task. If anything changes during plan execution, you repeat the SMPA cycle as required.

A key concept in AI is the planner as an automatic task designer. The planner designs the target task by using mathematical logic to select the best task from a suite of reasonable or possible task designs. Some of the first AI planners such as the General Problem Solver described by Newell, Shaw and Simon in [5] were based in part on theorem proving programs, such as the Logic Theorist also by Newell, Shaw and Simon in [6]. In

this view, designing a plan is reduced to an algorithmic process, like solving an equation. For a discussion of AI planning and its problems, see Brooks [7] and Pollock [8]

In practice, AI planners are less automatic than intended. As Pollock [8] points out in reference to the STRIPS planner developed by Fikes and Nilsson in [9], "... What is important here is that the planning is interleaved with epistemic reasoning aimed at finding appropriate information for use in the planning, and the course of the epistemic reasoning is directed by where the agent is in its planning. In effect, requiring the human operator of an AI planner to perform this epistemic reasoning in advance requires him or her to already know how the planning problem will be solved, but this makes the planner superfluous." In this view, AI planners look more like pre-designed plan repositories than independent plan designers.

The AI model depends on a model of the world and the robot within it. Given complete knowledge of the world the robot inhabits, such a model can be manipulated to generate plans for tasks. Industrial robots in factories occupy this kind of closed, known world, and their performance has been good as a result. This model also works well in spacecraft robots. All reasonable failures and failure combinations can be programmed into the internal model of the craft, and an action is defined for each failure combination. All spacecraft states are therefore known, and the appropriate action is defined for each state.

However, the AI model does not work well in open environments, where knowledge of the world is inherently incomplete. AI uses logical models of the world to predict the future from the state of the current model and use these predictions to form a plan. In an unpredictable world, this is either logically impossible or computationally intractable. A variant of this problem is the famous frame problem, discussed by Ford and Pylyshyn in [4] which is the problem of predicting what does not change in the world when the robot changes some thing. For a discussion of other problems with the AI approach, see Pollock [8], Ford and Pylyshyn [4] and Brooks [3]. The AI approach does not work in open environments, so it fails our first requirement.

The AI approach also does not have an inherent concept of agency. AI implies independence. As a synthetic human, an artificially intelligent machine should independently think for itself. There is no direct path to the inclusion of agency. The tacit assumption seems to be that once we learn how intelligence works, we will be able to design a robot that will naturally act as an agent for us. Incidentally, the concept of planning and using planners supplies an indirect tradition of agency. Planners require goals as inputs to the planning. We do not understand AI well enough to model it as a useful generator of these goals, so human users typically supply them. The robot is acting as an agent, but only temporarily to support development.

3.3 Behavior-Based (BB) Model

The Behavior-Based (BB) model focuses on creating desired behavior in a robot. Arkin in [10] defines behavior as: "A behavior is (...) a response to a stimulus." Robot behavior in an environment is a response to stimulus from the environment as perceived by the robot. This can be an outside stimulus such as a light or sound, or it can be a stimulus caused by

the robot sensing an obstacle as the robot approaches it. The BB model is a mapping of stimuli to responses. It has evolved along with the AI model, with both having origins at around 1950. The BB model is associated with cybernetics, the study of communication and control in man and machine, as originally defined by Weiner in [2].

BB robots can be divided into two categories. The first category is reactive systems and contains pure stimulus-response BB robots that have neither state nor planning. The second category is hybrid systems and adds state and planning elements to the reactive systems model.

Perhaps the first example of a BB robot was the W. Grey Walter's mechanical tortoise described by Walter in [11]. It looked a little like a tortoise and moved around the floor on three wheels, a single wheel for driving and steering, and two idling wheels. It had two sensors (a photocell and a bump sensor) and two actuators in the form of motors, one for the drive wheel and one for steering. It was a simple robot with three behaviors, but it demonstrated what appeared to be sophisticated, even goal oriented activity. Simple behaviors and a complex environment produced complex activity.

In reactive systems BB robots, behaviors are designed in and selected at run time to produce the desired robot activity. Artificial neural networks have been used to allow BB robot behaviors to be trained-in rather than designed-in, but the intent is the same. Such robots resemble mechanical animals or insects, particularly those with multiple legs. Indeed, animals and insects are studied for ideas in behavior that can be applied to BB robots. This is called ethologically guided or constrained design. See Brooks [7] and Pollock [8] for a more thorough discussion of BB robotics.

However, the reactive systems BB design model does not have the problem with open environments that the AI model has. It reacts directly to the environment as it experiences it, whether open or closed. It has no model of the environment to be incomplete or in error. It is not bothered by unpredictability because it makes no predictions. This meets our first requirement of being able to operate in an open environment.

The reactive systems BB model does not include agency, as we have defined it. As a synthetic animal or insect, it is expected to think for itself. It maps stimuli to responses with no memory other than the mapping itself. Since it has no state memory, there is no way to give it a task to remember or ask it about tasks completed. Current floor-cleaning and lawn-mowing robots are of this type. However as Brooks in [3] points out, the BB model can certainly have state and is not constrained to be purely reactive. For example, the Allen robot described by Brooks in [12] had a higher layer that would select a goal for the lower, reactive layer to achieve.

If the reactive systems BB model has no state memory, it cannot include our concept of agency, where agency is considered to be a real-time request for task performance.

3.4 Hybrid Model

Hybrid models are combinations of the deliberative elements of AI and the reactive elements of BB in an attempt to use the advantages of one model to compensate for the

deficiencies of the other. Many combinations are possible, from primarily AI to primarily BB. Arkin in [10] provides a review of many of these hybrids.

Combining the AI and BB models is problematic. The AI model relies on knowledge contained in a model in order to plan and generate actions, while the BB model does not have such a model. The AI model retains all the problems of rational planning, and there is no obvious way for the BB model to help this. A major problem of hybrid models is how to combine deliberative and reactive elements in a working design. As Arkin in [10] notes, "The nature and the boundary between deliberative and reactive execution is not well understood at this time, leading to somewhat arbitrary architectural decisions." Also, planning that uses behaviors will be much different than conventional AI planning, as pointed out by Brooks in [7] and by Agre and Chapman in [13].

The intent of the hybrid model is to add planning to the reactive BB model. The goal is to provide task planning capable of operating in an open environment. Assuming that the hybrid model is based on reactive behavior, it should be able to operate in open environments, meeting our first requirement.

Like the AI and the reactive systems BB models, the hybrid robots have no direct concept of agency. Both assume that the robot is an independent, synthetic animal or insect that thinks for itself, violating our requirement for agency.

4 The Animal Training (AT) Model

The Animal Training (AT) model provides us with an image and a design methodology for a robot that can potentially meet our design requirements. 1) Because it is behavior based, it will work in open environments. 2) The human designer/user does the task design, providing a direct model of agency.

4.1 Robot Design Using the Animal Training Model

In the AT model, the robot designer is in the position of the dog trainer, and the robot user is in the position of the shepherd. Both are involved in the robot design, but at different points. The robot designer creates a robot with the physical characteristics and behavior capabilities useful for its range of tasks in its set of environments. The robot user then designs task programs that use these primitive behaviors.

User programming of the robot is significant because a robot is an agent designed to do tasks for humans in open environments. All open environments are unique but with discoverable similarities. For example, even though two apartments may be identical in layout, the owners will furnish them uniquely. However, both apartments are similar because they will have sofas, chairs and tables, although different in type and style. As a result, each robot task design is unique to its intended open environment. Each user must design the tasks for the robot's unique environment.

Human task design for robots is finite in scope in the same sense that dog training is finite in scope. The robot is not expected to deal with all possible situations, just the typical ones. If it encounters a situation that it cannot deal with, it calls for help. An independent entity does not have this option; in principle, there is no one to call. As a result, human task design starts with a few practical tasks and the ability to deal with a few typical situations, such as avoiding an obstacle in the path to a goal. Assuming tasks can be added and extended incrementally, the robot can become subtle and sophisticated simply by accretion.

Robots in the AT model are not independent entities. They have no beliefs, desires or intentions of their own. Since robots in the AT model are not modeling independently acting humans or animals, there is no requirement for their design to be biologically plausible. However, animals and humans can be useful sources of design ideas.

4.2 Robots as Pick-and-Place Machines

Robots are designed to do tasks, and their tasks and environments largely determine their design. A house robot, an undersea robot and a planetary explorer robot are significantly different in their designs because of their very different tasks. However, they are all robots, so it is reasonable to ask what design characteristics they have in common.

As a broad generalization, robots are pick-and-place machines designed for open environments. Pick-and-place is a term used in industrial robotics to indicate that the robot picks up something from one location and puts it in another. Picking up includes manipulation of the object selected. Exploration robots are pick-and-place robots that seldom actually pick up something. Their job is to find interesting things, things that would be worthy of picking up. Manufacturing, cleaning and maintenance robots extend the definition of the picking-up to include manipulation of the things chosen for pick-up.

The combination of the expected task and its environment determines what is picked up and where it is placed. A house robot may get you a soda from the refrigerator, then pick up and put away the children's toys. A construction robot may unload wood from a truck and distribute to building sites. A planetary rover robot may explore an area looking for candidate objects to pick up, or to simply identify.

Since this is a broad generalization, there will be robots that do not seem to fit the model of pick-and-place. An example would be a social robot, such as a museum guide. However even in these cases, their implementation may be a subset of the pick-and-place model due to the need to work in open environments, for agency and for communication.

4.3 Pick-and Place-Requirements

Pick-and-place operations require several activities: 1) The robot must move to where the object for pick up is to be found, 2) It must search for and identify the object. 3) It must move to the object and pick it up, and 4) It must carry the object it to its destination and

place it there. These operations require movement, navigation, object recognition, object grasping, pick-up, place and release.

A pick-and-place robot can be described as a robot arm with a platform to move it around and a vision system to guide it. The first requirement is a platform that can move around efficiently in the robot's intended environments. This is a challenging mechanical engineering design problem, one that can be simple in definition but complex in execution.

The robot needs to navigate relative to the objects in its environment because it needs to either go to them or avoid them. This kind of navigation is called pilotage, as in piloting a ship in a harbor. Pilotage is required for maneuvering in an open environment where objects can move around. Docking is precision pilotage, as in bringing the boat next to the dock. Docking is required for object manipulation, such as picking things up.

Robot pilotage requires recognizing objects (e.g. landmarks) and determining their location. It is also required for finding and picking up objects. Recognition of specific objects may not be required for collision avoidance, but is helpful even then. Because object recognition and location is central to our pick-and-place robots, their design can be called Object Based Design (OBD).

As an example of a pick-and-place house robot, consider a robot that will get you a soda from the refrigerator in the kitchen and bring it to you as you sit on the couch. To begin with, the robot must go to the kitchen. By recognizing surrounding landmarks, it can determine its location and select a path that will take it to the door into the kitchen and into the kitchen itself. Then, it will look for and recognize the refrigerator as another landmark. The robot will proceed to the refrigerator and dock in front of it. Then, it will search for and recognize the door handle. The robot opens the refrigerator door, searches for a can of soda, recognizes and locates the can, guides the arm and gripper to the can and picks it up. The robot then closes the refrigerator door, navigates back to you and hands you the soda. As this shows, object recognition is essential for pick-and-place operations.

4.4 Object Recognition

Object recognition – particularly using vision - has been an area of intense study for decades. The results of these efforts have tended to be limited and specific rather than general. A recent advance offers to change this situation. The advance is a new vision based recognition algorithm, the Scale Invariant Feature Transform (SIFT) developed by David Lowe, as described in [14], [15] and [16]. This transform can uniquely recognize objects at any distance or pose and with varying lighting and occlusion. This capability is what the object recognition community has been seeking.

The SIFT algorithm can identify an object and measure its location in the image. Given knowledge of the camera optics and direction of gaze, the object bearing (angle to the robot) can be found. Using stereo vision or other distance measuring techniques, the range of the object can be found. The result is an identification of objects in the field of view and their location relative to the robot. Given algorithms like the SIFT algorithm that can provide rapid and reliable identification and location of all objects in view, pilotage, docking and grasping are straightforward engineering problems.

4.5 A Robot Architecture Using the Animal Training (AT) Model

The AT robot design model consists of a robot with useful behaviors and tasks programmed by a human user and that executes those tasks on verbal command. Physically, it is a mobile platform carrying a camera for object recognition and an arm for grasping and manipulating.

A robot architecture that can implement this model consists of a 1) a behavior manager, 2) a task manager and 3) a dialog manager. The behavior manager processes sensor data to drive actuators, such as the motors in the wheels (or legs) and arms of the robot. The task manager converts behaviors into tasks that achieve a goal, and it selects and executes tasks given to it by the dialog manager. The dialog manager is the interface between the human user and the task manager. It communicates with the task manager to select tasks and to create new tasks.

The behavior manager uses the identities and relative locations of objects to control the various actuators of the robot, as determined by the active behaviors. It consists of sensor processors (e.g. the vision system) to identify and locate objects, an analog logic system as described by Wyland in [1] to convert this information into actuator control signals, and actuator processors for local control of actuator motors.

The task manager executes tasks previously supplied by the human user. In this paper, we will assume the RAP model described by Firby in [17] for behavior based task management. Tasks consist of primitive tasks and composite tasks. A primitive task is a behavior that achieves a goal. The behavior is selected and enabled by the task manager. The task manager also determines when the task is done and whether it was successful. A composite task is a tree structure of sub-tasks and primitive tasks.

Because the robot operates in an unpredictable environment, tasks can succeed or fail, and they can succeed in more than one way. If a task fails, the task manager must know what to do. An optional retry list for each task can serve this function. If a task fails, the task manager tries the next task on the retry list. If it reaches the end of the list without success, it passes the failure back to the task that called it or notifies the human at the appropriate time. This notification is a task of its own. Retry lists are often used by humans. For example, you may have a list of things to try if your car will not start.

The dialog manager provides structured natural language communication between the task manager and the human user. The robot only contains what we put in it. We enter the phrases and sentences we want it to recognize, and we enter the responses to those sentences. In concept, this is much like a frequently asked questions (FAQ) list. The difference is that the answers will reflect the current state of the robot, such as tasks completed. A simple conversation processor such as described by Suerth in [18] can be used as the dialog manager. Although the robot may eventually acquire a large number of input sentences and their responses, this is not as daunting as it may seem. The robot only needs to recognize the sentences that initiate its tasks, including replying to status questions. Sentences and their responses are added incrementally as needed for new tasks. Initially, only a small number of sentences and tasks may be required for the robot to be useful. This number can grow easily and incrementally with time. Eventually the robot may be capable of sophisticated tasks and conversation by simple accretion. In effect, the

robot will contain the complete policy manual for a perfect clerk, where every useful, relevant question or request will be recorded, along with its appropriate response.

The initial dialog processor can be purely text based. This simplifies the recognition problem. Speech synthesis can be added for a verbal response, and speech recognition can be added for verbal input. Speech recognition – typically a difficult problem – can be aided by the finite nature of the vocabulary and dialog.

The dialog manager includes an episodic memory. This memory records task information from the task manager, such as tasks completed and objects recognized as part of task execution. The episodic memory remembers tasks executed and information specific to the task. Note that scanning for objects is a task. This limits and defines the scope of the memory. The episodic memory allows status reporting and history analysis, which is useful in task design and debug.

4.6 Task Design by Giving Directions

Robot design is about task design. In the AI and hybrid models, tasks are supposed to be designed automatically. In the BB model, tasks are designed-in as primitive behaviors. In the AT model, tasks are designed by the robot users.

The ordinary activity of giving directions is a form of task design, also known as task planning, as pointed out by Agre and Chapman in [13] and by Payton in [19]. Directions are commonly given for navigation purposes, as a task plan for going somewhere from a known starting point. Consider the task “Get me a soda.” Detailed directions for this task might be: 1) Go to the kitchen, 2) Go to the refrigerator, 3) Open the refrigerator door, 4) Find a soda, 5) Reach in and grasp the soda, 6) Pull the soda out of the refrigerator, 7) Close the refrigerator door, 8) Return to me (the requestor), 8) Present the soda to me, and 9) Release the soda can when I (the user) grasp it. These directions assume that each step (i.e. subtask) is a task that the human (or robot) knows how to do.

Directions are object based. Each step of these navigation directions refers to an object as the focus of the task. Most steps refer to movement from the current location toward a named object or a named place defined by objects around that place. Other steps involve manipulation of a target object, such as opening the refrigerator door and grasping the soda. The implied primitive tasks for these directions could be movement from the current position to a position near an object; grasping, opening and closing the refrigerator door; and grasping, picking up and releasing the soda can.

4.7 Task Memory and Task Selection

In the Animal Training (AT) model, the user selects previously designed tasks. This is planning by remembering, as described by Hammond in [20]. Tasks are named, and the dialog manager can select tasks to be done by name. Task design by giving directions implies each step of the directions is a subtask with a named object-based activity as its target. The task manager includes a task memory that holds the task designs.

Planning by remembering in the AT model does not mean the rejection of other forms of planning. In the process of creating tasks and using the robot, patterns of use and programming will develop. These patterns can be exploited to support portions of the design process.

4.8 Building the Robot – Work in Progress

We are building a robot based on the Animal Training design model as described in this paper. The first prototype is a battery powered, autonomous, indoor robot. It consists of a 12” square platform of a conventional design with two driven wheels plus a caster. The platform holds a small robot arm, a video camera and an embedded PC motherboard running Linux. It will be used to integrate the AT elements in a complete working robot.

5 Summary

The Animal Training (AT) design model allows us to design robots as articulate machines that do tasks as agents for human users in open environments. It provides an inherent model of agency, with a formal separation of human provided goals and robot behaviors. Neither consciousness nor intelligence is required or even meaningful because the robot is programmed as an autonomous machine, not an independent entity.

The AT model is behavior based, allowing the robot to function in an open, unpredictable environment. Its behaviors are designed-in, and the human user programs all its tasks. The task design problems of AI are avoided by relegating the designs to the human users. Object based behavior design and task design by giving directions simplify these designs. Robot task sophistication grows naturally by accretion. The result is an articulate, autonomous agent that performs tasks for humans in an open environment.

References

- [1] David C. Wyland, Reasonable Machines: Analogical Reasoning in Autonomous Agent Design, Innovative Concepts for Agent-Based Systems, *First International Workshop on Radical Agent Concepts (WRAC 2002)*, ed. Walt Truszkowski, Springer 2003, ISBN 3-540-40725-1.
- [2] Norbert Wiener, *Cybernetics*, Second Edition, MIT Press, 1948.
- [3] R.A. Brooks, Intelligence Without Representation, *Artificial Intelligence Journal* (47), 1991.
- [4] K. M. Ford and Z. W. Pylyshyn, *The Robot’s Dilemma Revisited*, Ablex Publishing Company, 1996, ISBN 1-56750-142-7.
- [5] Allen Newel, J. C. Shaw, Herbert Simon, “A General Problem Solving Program for a Computer,” *Computers and Automation* 8(7), 1959, 10-16.

- [6] Allen Newel, J. C. Shaw, Herbert Simon, "Empirical Explorations with the Logic Theory Machine," *Proc. Western Joint Computer conference*, 15, 1957, 218-329.
- [7] R.A. Brooks, "Intelligence without Reason," Proceedings of the 1991 International Joint Conference on Artificial Intelligence, pp 569-595.
- [8] John Pollock, "Planning Agents," ed. Rao and Wooldridge, Kluwer Academic Publishers 1999; ISBN: 0792356012
- [9] R. E. Fikes and N. J. Nilsson, "STRIPS: a new approach to the application of theorem proving to problem solving," *Artificial Intelligence* 2, pp 189-208.
- [10] Ronald C. Arkin, Behavior Based Robotics, MIT Press, 1998, ISBN 0-262-01165-4.
- [11] W. Grey Walter, "The Living Brain," W. W. Norton, New York, 1963
- [12] Rodney A. Brooks, "A Robust Layered Control System for a Mobile Robot," *IEEE Journal of Robotics and Automation*, RA-2, April, 1986 14-23.
- [13] Philip Agre & David Chapman, "What are Plans For?," *Designing Autonomous Agents*, ed. Pattie Maes, MIT Press 1990, ISBN 0-262-63135-0.
- [14] David G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, 60, 2 (2004), pp. 91-110.
- [15] David G. Lowe, "Object Recognition from Local Scale-Invariant Features," *International Journal of Computer Vision*, Corfu, Greece (September 1999), pp. 1150-1157.
- [16] Stephen Se, David Lowe and Jim Little, "Local and Global Localization for Mobile Robots Using Visual Landmarks," *IEEE International Conference on Intelligent Robots and Systems, IROS 2001*, Maui, Hawaii (October 2001), pp. 414-420.
- [17] R. James Firby, "Adaptive Execution in Complex Dynamic Domains," Ph.D. Thesis, *Yale University Technical Report YALEU/CSD/RR #672*, 1994.
- [18] Russell Suereth, Developing Natural Language Interfaces, McGraw-Hill 1997, ISBN 0-07-913017-8.
- [19] David W. Payton, "Internalized Plans: A Representation for Action Resources," *Designing Autonomous Agents*, ed. Pattie Maes, MIT Press 1990, ISBN 0-262-63135-0.
- [20] K. J. Hammond, "Case-Based Planning: A Framework for Planning from Experience", *Cognitive Science*, volume 14, pages 385--443, 1990