

Binary Agents for the Control of Autonomous Vehicles

Andrew Wallace
Frontios
AAWallace@iee.org

Abstract—Behavioural based systems are commonly used in robotic research. Arguably, the main area of activity has been in the area of integrating behavioural or reactive systems with deliberative systems in the form of hierarchical systems and in the area of action selection. This paper presents a novel architecture that is intended to lend itself to the action selection problem. The architecture centres around the use of binary agents. A binary agent is one that can only accept an ‘on’ or an ‘off’ signal as its input. Once on it then cycles through a set of states turning other agents on or off. A sub-class of binary agent is also used which uses an additional threshold value for activation. An example application is presented in the form of an autonomous mobile platform and a description of its use in experimentation is given. The architecture is then discussed before future work is presented.

I. INTRODUCTION

Since its inception in the mid 1980s [1], behavioural based robotic controllers have been shown to be a successful method for controlling autonomous platforms such as mobile robots. As a result, it has formed a very active area in robotics research (see [2] for a good overview of behavioural based robotics).

Behavioural or reactive based systems, however, have two main problems. The first is concerned with handling complex scenarios that require planning such as navigating to a goal position and the second is the construction of large and / or complex systems in terms of designing the mechanism of selecting the required behaviours as the system can become too complex to build by hand. Therefore, arguably, the two main areas that have been researched within the field of behavioural based robotics have been the combining of deliberative based architectures with reactive based architectures and the action selection problem.

Examples of combining architectures includes [3] who use layered architecture for an office delivery robot using reactive and a decision-theoretic methods for navigation and obstacle avoidance. Examples of the action selection problem include [4], [5] and [6] where evolutionary approaches are taken to evolve the action selection of a behavioural based system. In [7] and [8] learning momentum and Q-Learning are methods used for action selection. However, each method uses traditional types of behaviours, which have to be hand crafted. This paper, however, presents an alternative construction method for autonomous, behavioural based, vehicles such as mobile robots. The method centres around the use of binary agents and is intended to lend itself to the solving of the action selection problem by providing elementary building

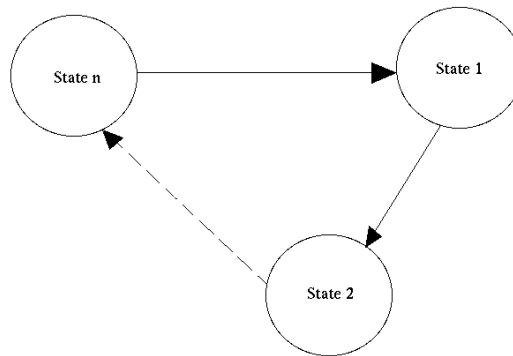


Fig. 1. State diagram for a binary agent

blocks, which through networking can form behaviours and a behavioural based controller. They are intended to form the foundations of a method to automatically construct behaviours for complex systems such as mobile robots, humanoid robots and information fusion systems.

This paper presents research into the application of such binary agents to the control of an autonomous vehicle. After presenting a description of the agents in section II this paper presents an example system that was implemented on a physical agent (that is, a mobile robot platform) consisting of a light sensor, touch sensors and two motor actuators in section III. The robot was then given the task of exploring an unknown environment. A description of the experimental run with the robot is given in section IV as well as a discussion in section V, summary in section VI, conclusion in section VII and future work in section VIII.

II. BINARY AGENTS

A. Agents

There are various definitions of “agent” in the literature such as:

“...embodied systems that behave in the real world without human control.”[9]

“...a system that tries to fulfil a set of goals in a complex, dynamic environment... An agent is called *autonomous* if it ... decides itself how to

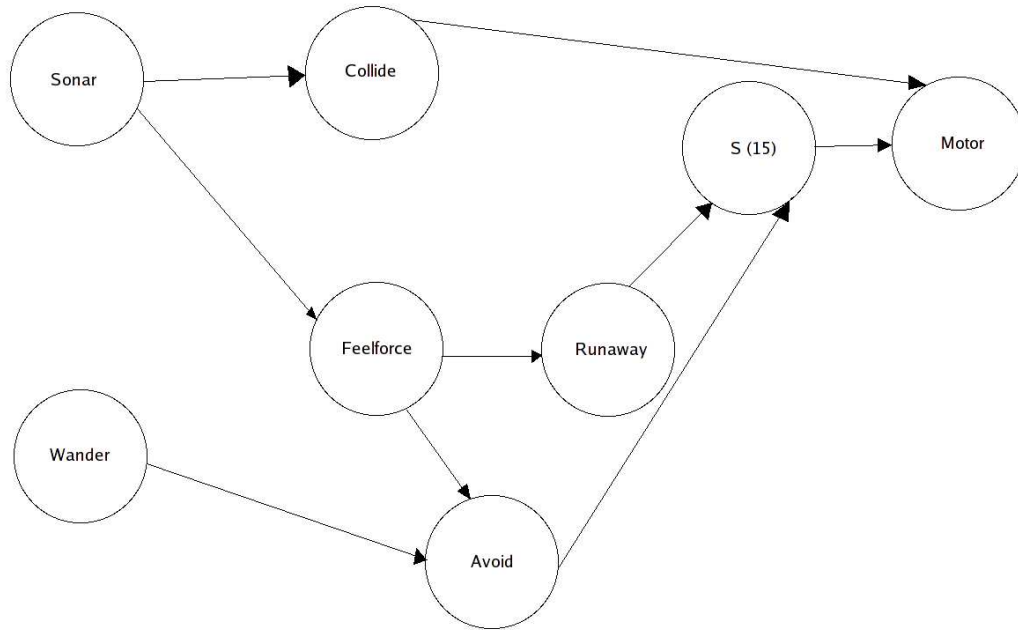


Fig. 2. Layer 0 and 1 of Brook's subsumption architecture seen as a digraph

relate its sensor data to motor commands in such away that its goals are attended to successfully. An agent is said to be *adaptive* if it is able to improve over time.” [10]

However, for this paper an agent is defined as:

“A task orientated entity that performs one, well defined, task at a specific level of abstraction.”

A well defined task is defined from the perspective of the agent designer and to achieve the overall system goal a group of agents, referred to as a society of agents, will most likely be required. The inputs to the agents can be direct sensor inputs, signals or messages from other behaviours.

B. Binary Agents and Threshold Binary Agents

The simplest form of agent would be one that takes a single input of a binary value. Such an agent is referred to as a binary agent.

A binary agent, therefore, is a sub-type of agent that consists of a finite state machine augmented with timers, which is activated by a binary on / off signal. That is, binary agents can only accept inputs in the set $A = \{0, 1\}$. The binary agents are then connected in a network of agents to form a controller. Any one agent can be connected to one or more other agents. It follows, therefore, that for every state that the agent is in it will be able to send a binary signal to 0 or more agents, thus turning them on or off. The pattern of signals sent is referred to as the activation pattern and can be different for each state.

For example, if AgentA, AgentB and AgentC were binary agents so that AgentA was connected to both AgentB and AgentC. AgentA could turn AgentB on and AgentC off in one state. Its activation pattern for that state would then be 1, 0. In its next state it could turn AgentB off and AgentC on. The activation pattern for that state would then be 0, 1.

Taking an object oriented view, a binary agent can be viewed as a class, which inherits from a more general class of Agent, and consists of a set of states augmented with timers defining its behaviour so that when an agent has been turned on it advances from state to state after a time-out. States are, therefore, arranged in a circular pattern looping from end to start. Thus the transition from state to state is an ordered sequence, $S = \langle 1, 2, \dots, n \rangle$ as shown in figure 1.

A behavioural based system is a layered architecture that couples sensory input directly to actuators. However, it can also be viewed as a digraph where each behaviour can be viewed as a vertex and the interconnection between behaviours as arcs on a digraph (see figure 2).

Each behaviour within the system can be considered to be a finite state machine augmented with timers and thresholds. A binary agent approach takes this digraph view of a behavioural based system and defines each behaviour as one or more binary agents where each binary agent forms a vertex of the digraph, connected to other binary agents to form the robot controller.

The finite state machine can be considered also to be a digraph, where the states are vertexes and the transitions from state to state are the arcs. Thus, a binary agent system can be seen as a system of digraphs within digraphs.

A sub-class of Binary Agent is the threshold binary agent, which is a binary agent augmented with thresholds. In such a

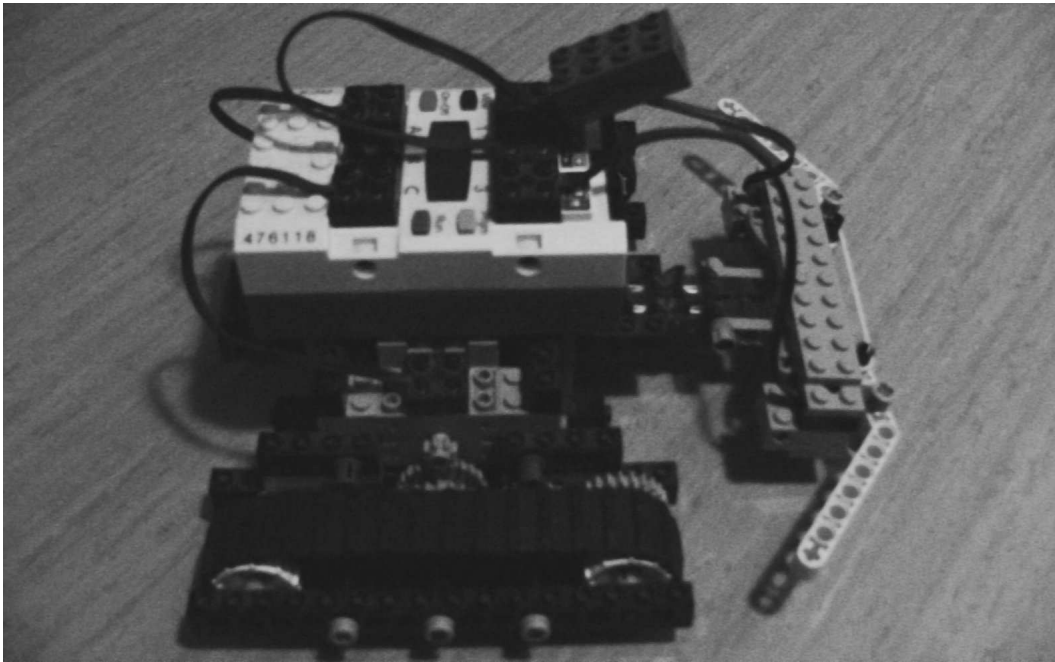


Fig. 3. The mobile robot platform

threshold binary agent the agent can be turned on either by an input value exceeding a threshold or a combination of an input exceeding a threshold and an on signal from another agent.

III. AN EXAMPLE SYSTEM

To evaluate the effectiveness of the agent based system, a controller was implemented utilising binary agents. The controller was designed to control a small mobile robot platform. This section gives an overview of the robot and the agent controller.

A. The Mobile Robot Platform

The small mobile robot platform used was a Lego robot consisting of two motor actuators (referred to as motor A and motor C), two touch sensors (sensor 1 and sensor 3) and a light sensor (sensor 2) and is shown in figure 3. The two touch sensors were binary but the light sensor was digital and outputted values ranging from 0 to 255 which corresponded to a graded scale of light intensity from dark to light. The robot had three inputs labelled 1, 2 and 3 and three outputs labelled A, B, and C. The light sensor was connected to input 2 and a touch sensor was connected to the remaining inputs. The motors were connected to outputs A and C (hence the nomenclature give above). The robot was given the tasks of exploring an unknown environment and moving towards a light source.

B. The Binary Agents

To achieve its tasks the robot controller required agents to drive the robot forward, backwards and to turn so the robot was able to manoeuvre. It then required agents to be able to

detect the light level for searching for a light and agents to detect the activation of the touch sensors.

For the motion of the robot four agents were used. It can be considered that there are two well defined tasks for each motor; rotate clockwise and rotate anti-clockwise. One direction corresponds to forward and one to reverse. Turning is achieved by driving one motor in the forward direction and the other in the reverse direction.

Similarly, there are two tasks for the light sensor. One is to detect the light source and one is to detect the absence of the light source. This can be achieved by utilising two agents; one for detecting “dark” and one for detecting “light”, where “dark” and “light” correspond to a threshold value.

There are two tasks associated with the touch sensors. One is to detect the activation of the sensor and the other is to perform the required directional changes of the motors. Therefore, this leads to the following agents (see figure 4):

- Four motor agents that control the motion of the robot.
- Two light sensor agents to detect light thresholds
- Two touch sensor agents, one for each touch sensor
- Two additional binary agents to control the light sensors and motor agents when the sensor agents had been turned on.

1) *The motor agents:* Each motor agent had the task of turning the motor on in a given direction. Therefore, there were two agents for each motor; one to drive the motor forward and one for reverse as follows:

- MotorAFwdAgt
- MotorCFwdAgt
- MotorABkwAgt
- MotorCBkwAgt

The Motor*FwdAgt_s drove the respective motor in the forward direction and the Motor*BkwAgt_s the reverse direction.

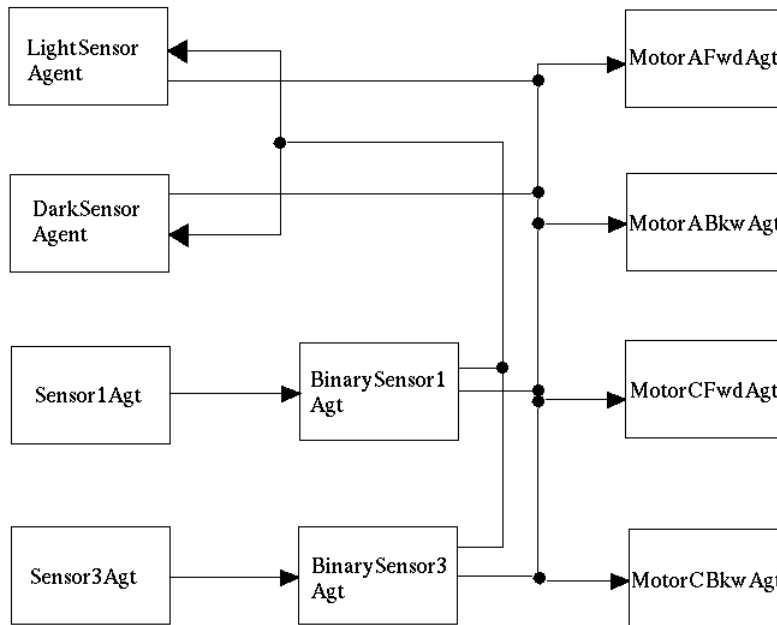


Fig. 4. Binary agents used in the robot controller

Each agent had only one state in which they switched their respective motor on or off.

2) *The light sensor agents:* To direct the robot to a light source there were two light agents. Each light agent was of the threshold type of binary agent. The DarkSensorAgent was turned on when the light level fell below a threshold and the LightSensorAgent was turned on when the light level went above a threshold level. The light sensor agents were connected to the light sensor and to the four motor agents. When each respective light agent was turned on they would turn on or off the motor agents. The DarkSensorAgent turned off one reverse motor agent and one on and one forward motor agent off and one on resulting in the robot spinning on the spot in a “search out the light” behaviour. When the light sensor agent was turned on it turned off both reverse motor agents and turned on both forward agents. This resulted in a “head towards the light” behaviour. Each agent had one state.

3) *The touch sensors:* The touch sensor agents (Sensor1Agt and Sensor3Agt) were directly connected to the touch sensors and to two other agents. The two other agents were referred to as BinarySensor1Agt and BinarySensor2Agt. The sensor agents had the task of turning the binary sensor agents on when the touch sensor was closed. The BinarySensor*Agts had the task of turning off the light sensor agents and the motor agents. As the binary agents cycled through their states they first turn both reverse motor agents on and both forward agents off causing the robot to reverse. They then turned one forward motor agent on causing the robot to rotate. Finally they turned both forward agents on causing the robot to move forward. The light sensors were then also turned on. This resulted in a “move away from object” behaviour which occurred after a collision with an object.

The Sensor*Agts each had one state. The BinarySensor*Agts had four states. The activation patterns used in each

of the four states is shown in table I.

IV. EXPERIMENTATION AND RESULTS

To evaluate the controller the robot was given the task of exploring an unknown environment. The environment was a flat surface with a random set of object placed at various locations and was surrounded by a wall on all sides. The robot then explored the environment colliding with objects. As it did so the expected set of behaviours were observed. The robot reversed, turned and then advanced after each collision.

After the robot ran for one minute the light sensors were activated. When a light source was not present or detected the DarkSensorAgent was active causing the robot to enter into a light seeking behaviour. When a light source was detected the LightSensorAgent was active causing the robot to enter into the “head towards a light” behaviour.

If a collision was detected the light sensor agents were turned off and the robot went into the “move away from object” behaviour.

The robot performed as expected with the switching of external observable behaviour achieved by switching agents on or off.

V. DISCUSSION

The binary agent controller was able to control a mobile robot platform and externally observable behaviours were evident despite there being no explicit behavioural representation being programmed. The resulting behaviours can be seen as emerging from the interaction of the binary agents or they can be viewed as encoded in the states of the binary agents and the activation patterns and the network of interconnected agents.

The application and the construction of the agents and their states were engineered by hand. As the agents are formed in

State	MotorAFwdAgt	MotorABkwAgt	MotorCFwdAgt	MotorCBkwAgt	LightSensorAgent	DarkSensorAgent
1	off	on	off	on	off	off
2	on	off	off	on	off	off
3	on	off	on	off	off	off
4	on	off	on	off	on	on

TABLE I
STATE ACTIVATION PATTERNS FOR BINARYSENSOR1AGT

a network where activation signals are sent from one agent to one or more other agents it can be considered to be similar to a network of neurones in a Artificial Neural Network. The activation pattern in such a network can be learnt as well as the interconnections and the network itself. Therefore, there is a possibility that the state machines, the on / off activation patterns for each state and the network of binary agents could be learnt. Methods such as reinforce learning or genetic algorithms could be employed to learn the structure of a binary agent controller.

Another possibility for the automated formation of binary agent controller and / or the state machines and activation patterns is the use of cellular automata. Cells in a cellular automata network are either on or off and as such have a binary state similar to binary agents. The rules for determining the on or off states for each cell could be used to decide the connections between the binary agents or the activation patterns and state machines of each agent.

Other aspects of interest to study is the scalability of the system which is associated with handling more complex problems. Also it would be interesting to study how binary agents would handle noisy or imperfect data.

The demonstrated system presented here was intended to be a simple system used to present the basic ideas but would binary agents be able to handle more complex problems where the system could potentially run into hundreds, thousands or more agents? Could such complex systems be developed using reinforced learning or genetic algorithms?

The sensors used were relatively noise free. The touch sensors were either on or off but there was some possibility of noise on the light sensor. How would the binary agent handle noisy inputs? How would they handle variations in thresholds (such as the background lighting where the "dark" and "light" threshold values may change from room to room, for example)?

VI. SUMMARY

This paper has presented binary agents. A binary agent can be considered to be a sub-class of a more general agent class where the binary agent can only be turned on or off. The behaviour of a binary agent is defined by a circular finite state machine. For each state that the agent is in it is able to send an on or off signal to one or more binary agents. In addition to the basic binary agent a threshold binary agent was also presented. This is a binary agent augmented with a threshold so that the agent can be turned on only if an additional input is above or below a given threshold.

A binary agent controller is then constructed of a network of binary and threshold binary agents. An example controller

was presented. The controller was designed to control a mobile robot so that it could accomplish two tasks. The first was exploration of an unknown environment and the second was to seek out a light source. A number of externally observed behaviours were noted. Their behaviours did not have any corresponding behavioural modules in the controller.

VII. CONCLUSION

Binary agent present a possible means for controlling an autonomous vehicle such as a mobile robot. They have the potential to be automatically generated using such methods as cellular automata, reinforced learning or genetic algorithms. However, there is still additional work needed to be undertaken to investigate their possibilities and there remains a number of potential problems that could limit there usefulness.

VIII. FUTURE WORK

Future work in binary agents will concentrate on the automated construction of the agents' state machines, activation patterns and networks of such agents. It is intended that a number of different methods will be employed to ascertain which, if any such methods are applicable. Methods that could be used are genetic algorithms and reinforce learning of some type. A probable strategy would be to derive the state machines first and then the network, treating each as a separate problem. Next would be the investigation of deriving both state machines and network together.

Another method that could be investigated is cellular automata. Each cell could be a binary agent and the selection for the architecture could be dependant on rules.

Beyond that, additional work will need to be conducted into the scalability of a binary agent system and methods of implementation. For that, more complex tasks will need to be attempted.

IX. ACKNOWLEDGEMENT

The author would like to acknowledge the assistance given in producing this paper by Mrs E Wallace BSc.

REFERENCES

- [1] R. A. Brooks, "A robust layered control system for a mobile robot," Tech. Rep. A.I. Memo 864, Massachusetts Institute of Technology Artificial Intelligence Laboratory, September 1985.
- [2] R. C. Arkin, *Behavior-Based Robotics*. MIT Press, 1998.
- [3] R. Simmons, R. Goodwin, K. Z. Haigh, S. Koenig, and J. O'Sullivan, "A layered architecture for office delivery robots," in *First International Conference on Autonomous Agents*, pp. 245-252, February 1997.
- [4] M. Wahde and H. Sandholt, "Evolving complex behaviours on autonomous robots," in *Proceedings of the 7th UK Mechatronics Forum International Conference*, 2000.

- [5] M. Wahde, "A method for behavioural organization for autonomous robots based on evolutionary optimization of utility functions," *Journal of Systems and Control Engineering*, vol. 217, pp. 249–258, 2003.
- [6] C. Jassadapakorn and P. Chongstitvatana, "Reactive planning with evolutionary computation," in *National Computer Science and Engineering Conference*, pp. 357–361, 2002.
- [7] E. Martinson, A. Stoytchev, and R. Arkin, "Robot behavioral selection using q-learning," in *In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), EPFL, Switzerland, 2002*.
- [8] J. B. Lee and R. C. Arkin, "Adaptive multi-robot behavior via learning momentum," in *In Proceedings of IROS'03, 2003*.
- [9] R. Salomon, "Neural network in the context of autonomous agents: Important concepts revisited," *Artificial Neural Networks in Engineering*, 1996.
- [10] P. Maes, "Modeling adaptive autonomous agents," *Artificial Life Journal*, vol. 1, no. 1 & 2, pp. 135–162, 1994.