MALAYSIAN SCIENCE & TECHNOLOGY CONGRESS '98
Symposium C: Computer Science & Information Technology
Universiti Sains Malaysia, Pulau Pinang, 10 -11 November 1998

DISTRIBUTED BEHAVIOR-BASED CONTROL ARCHITECTURE FOR A WALL CLIMBING ROBOT

Nadir Ould Khessal

Faculty of Electrical Engineering
University Technology Malaysia
PO. Box 791, Skudai 80990
Johor, Malaysia
Tel: (60) -07-5505138
Fax:(60)-07-5566272
nadir.ok@ieee.org

Shamsudin H.M. Amin

Faculty of Electrical Engineering
University Technology Malaysia
PO. Box 791, Skudai 80990
Johor, Malaysia
Tel: (60) -07-5505003
Fax:(60)-07-5566272
sham@suria.fke.utm.my

ABSTRACT

In the past two decades, Behavior-based AI (Artificial Intelligence) has emerged as a new approach in designing mobile robot control architecture. It stresses on the issues of reactivity, concurrency and real-time control.

In this paper we propose a new approach in designing robust intelligent controllers for mobile robot platforms.

The Behaviour-based paradigm implemented in a multiprocessing firmware architecture will further enhance parallelism present in the subsumption paradigm itself and increased real-timeness.

The paper summarises research done to design a four-legged wall climbing robot. The emphasis will be on the control architecture of the robot based on the Behavior -based paradigm.

The robot control architecture is made up of two layers, the locomotion layer and the gait controller layer. The two layers are implemented on a Vesta 68332 processor board running the Behaviour-based kernel. The software is developed using the "L" programming language , introduced by IS Robotics.

The Behaviour-based paradigm is outlined and contrasted with the classical Knowledge-based approach. A description of the distributed architecture is presented followed by a presentation of the Behaviour-based agents for the two layers.

KEY WORDS

Legged robots, Behavior-Based AI , "L language", Artificial Intelligence.

1 INTRODUCTION

Since its introduction in 1986 subsumption architecture has been the subject of many research programs. It was first introduced by Professor Rodney Brooks in his landmark paper "A robust layered control system for mobile robot" [1]. It came as a result of several attempts to achieve robust real- time control systems.

With the increasing usage of robots in unstructured and hazardous environments such us planetary exploration, traditional AI showed a lack of real-timeness, and an increase in complexity, computing power and memory as the robot tasks multiplied.

Even though tremendous advances have been achieved with the introduction of fast VLSI computers, the traditional top-down architecture still could not result in robots that respond on time in a dynamically changing environment.

This is due primarily to the decomposition of the control system itself. Information about the external environment needed to travel through several layers before an action is taken (sense-think-act cycle). This resulted in power-hungry, time-critical processes such as perception , reasoning and planning.

Behavior-based paradigm argues that robots need not represent the world by constructing maps and generating paths for the robot to follow. However as argued by reactivists (as opposed to traditional AI followers [24] , [25], [32]) to respond to the dynamically changing world, the best world representation is the world itself [3], [9],[17],[19], [22], [28] and [23]. Sensory information needs to result in fast actions by being directly coupled to actuators.

Representing the world has never been an easy task, nor a perfect "shadow" of the actual world. Many attempts by traditional AI to symbolically represent the world required huge computing power and resulted in complex and difficult to maintain and debug systems.

Reactivists argue that to achieve robustness and real- timeness, simple primitive behaviors need to be built, on top of which other behaviors for more advanced (intelligent, autonomous) systems can be added later. It has been demonstrated that complex tasks could be accomplished using this control method by simply connecting sensors to actuators through simple machines using little or no internal state (memory).

Several actual designs based on the subsumption architecture have been demonstrated e.g. Genghis, Hanibal ,Herbert Squirt and Allen [4], [11], [12], [13] , [14] , [15], [16], [6], [5] and [7].

In this paper we will investigate the use of Behavior-based paradigm to design the control architecture of a wall climbing robot.

2 AN OVERVIEW OF BEHAVIOR-BASED CONTROL

As an alternative to the top-down model-based functional decomposition of the robot control system, the subsumption architecture is a bottom-up behavioral decomposition.

The Behavior-based control decomposes the control system onto task achieving layers [2] which consist of simple computational modules.

The lower layers are designed first to achieve the very basic behaviors of the robot (walk, move leg, ...). This is the layer nearest the robot mechanism where sensors are tightly coupled to the actuators, through simple processes. The robot is able to perform tasks for which this layer is responsible, without having to go through the conventional [sense-think-act] cycle.

Other layers of higher competence are then added on the top. Each added layer subsumes the lower layer , which in turn runs continuously, unaware of the higher layers.

Following the tremendous initial success of the subsumption paradigm, several research groups have progressed the subsumption paradigm.

In the MIT Artificial Intelligence Laboratory , subsumption architecture has resulted in robots of different sizes. Genghis [3] is a two-degree of freedom legged robot which can walk and climb over rough terrain. The subsumption based controller was built using a distributive multiprocessor architecture. Atilla [11] a later version of Genghis, is a three-degree of freedom legged robot, with every leg being associated with a processor.

Research groups elsewhere have used the subsumption paradigm and developed new architectures. Ronald C. Arkin from Georgia Institute of Technology developed Schema-based architecture named AuRA (Autonomous Robot Architecture) [18]. It is similar to the subsumption architecture in using reflexive behaviors called Motor-Schema. However it differs through its use of high level knowledge of the environment for the selection of motor and perception behaviors. It is based on reflexive behaviors , action oriented perception and homeostatic control [20].

In the Jet Propulsion Laboratory, researchers have used the subsumption architecture to design the control system of Planetary Rovers, Rocky III [28] and Rocky IV [27]. Recent work in the Research Center for Advanced Science and Technology, at the University of Tokyo, resulted in a reflection-based controller using inductive learning [21] .

Since the introduction of the subsumption paradigm several robot programming languages have been developed to implement different types of Behavior-based controllers.

As early as 1989 Brooks in MIT introduced the first subsumption compiler called the "Behavior Language" [8]. It is a rule-based real-time parallel robot programming language. It is used to compile Subsumption based programs and generate a machine code downloadable to several Motorola Processors (68000 and 68hc11) and Hitachi 6301. In addition it is able to generate a source program for Macintosh Common Lisp.

At the Artificial Intelligence Laboratory, University of Chicago, R. James Firby developed the "Chicago Robot Language " (CRL) a modular, Behavior-based distributed control system [30]. He later developed a similar language for the Animate Agent Architecture Project called "RAP Language " [31].

At the Jet Propulsion Laboratory, Erann Gat developed a language to construct reactive control mechanisms for autonomous robots and spacecraft, called the "Executive Support Language" (ESL) [29] .

ALFA (A Language For Action) developed in the same laboratory , unlike ESL and RAP, was designed primarily to describe reactive Behavior-based control mechanisms for autonomous robots [26] .

3 A BEHAVIOR-BASED CONTROLLER

The behavior language was written in LISP. In fact it is a subset of LISP. Real-time rules are the main component of the language and can be grouped to construct behaviors. The rules manipulate data available in registers, trigger and monitor mono-stables.

The "L" language is an advanced version of the Behavior language. It allows the use of Common LISP in real-time embedded systems. Besides using the previous processors as a target, it has also been designed as a stand alone product for Motorola 68332 based embedded systems.

It contains both a run-time environment and an interpreter which runs on the target system (Motorola Processor) and a compiler which runs on the host (Macintosh computer).

With these two pieces it is possible to develop a code on a host system (Macintosh) and download it to the target system. The run-time environment allows for interactive modification and monitoring of the target system during program execution[33]and [34].

The L system also contains two more pieces. MARS (Multiple Agency Reactive System) is added to make full use of Common LISP's Macro facilities and Venus is the interface between the L system and the hardware (Operating System).

3.1 The robot

The current platform is a modified version of IS Robotics Hermes II robot . Initially the robot walks using six legs, each with two degrees of freedom. It uses two forward-mounted whiskers for obstacle avoidance. It also incorporates force detectors on each leg, six infrared sensors as well as pitch and roll inclinometers.

Each leg is driven by two servo motors . The Swing servo motor enables the robot to perform a swing movement by rotating the leg forward and back The Lift servo motor used to lift and lower the leg.

3.2 The locomotion layer

Using the "L" development package we have implemented the lower control layer which we called the locomotion layer. The main behaviors implemented in this layer (figure 1) enable the robot to stick firmly to the wall as well as perform some primitive climbing movements such as move forward , move left and move right.

The development of the program is based on MARS (Multiple Agency Reactivity System) [10] a language for programming concurrent agents. It is a new version of the earlier Behavior Language .

In MARS, the definition of asynchronous parallel processes running in a single heap is done by defining assemblages (Behaviors) using the LISP-like form:

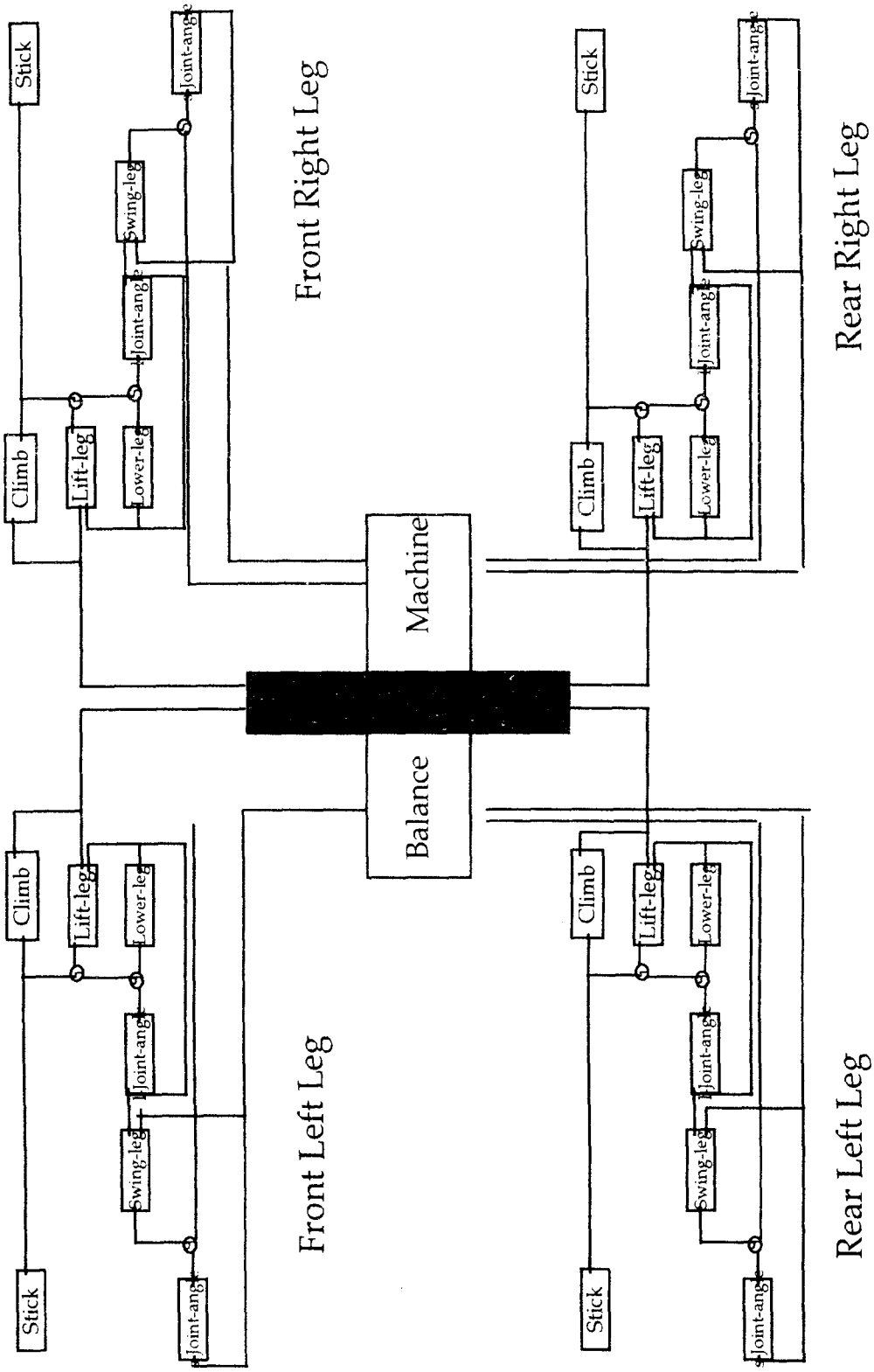

FIGURE 1. A Behavior Based Controller
The lower layer consists of parallel agents interacting through inhibition (i) and suppression (s) nodes.

The development of the program is based on MARS (Multiple Agency Reactivity System) [10] a language for programming concurrent agents. It is a new version of the earlier Behavior Language .

In MARS, the definition of asynchronous parallel processes running in a single heap is done by defining assemblages (Behaviors) using the LISP-like form:

def-assemblage-maker metaname
Lamda-list
descriptors
{form}

Whenever the produced procedure is called, it creates a an instance assemblage. An assemblage consists of ports , slots, monostables and concurrent processes running on a single heap.

Processes are spawned using one of the two macros, spawn-lwprocesses or spawn-hwprocesses, to define two types of processes, lightweight processes and heavyweight processes. The difference is that the heavyweight processes have their own stacks while the lightweight processes run in the L system stack.

spawn-hwprocesses *&key (whenever 't)*
wait waitticks rateperscond
(schedule :asavailable)
(stacksize '64) (ticks '4)
Pname body

spawn-lwprocesses *&key (whenever 't)*
wait waitticks rateperscond
(schedule :asavailable)
Pname body

Figure 2 shows a single leg locomotion layer which consists of seven behaviours namely Climb, Stick, Lift-leg, Lower-leg, L-joint-angle, S-joint-angle, Swing -leg.

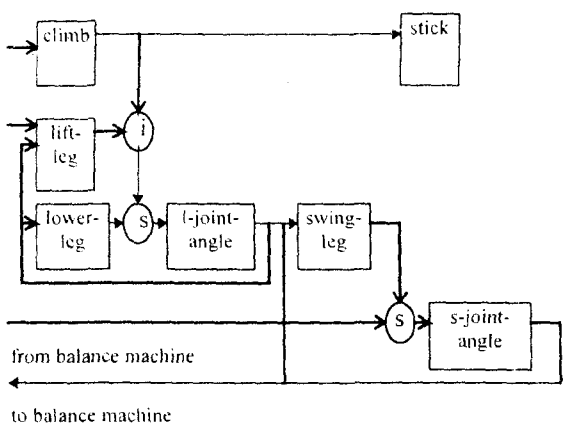

FIGURE 2. A Single Leg Locomotion Layer.

Joint-angle assemblage: this assemblage controls the joints of the legs. There are eight assemblage instances corresponding to the eight joints :

(joint-angle 'sfl) Swing Front Left
(joint-angle 'lfl) Lift Front Left
(joint-angle 'sfr) Swing Front Right
(joint-angle 'lfr) Lift Front Right
(joint-angle 'srl) Swing Rear Left
(joint-angle 'lrl) Lift Rear Left
(joint-angle 'srr) Swing Rear Right
(joint-angle 'lrr) Lift Rear Right

The assemblage instance joint-angle is defined by the assemblage-maker , joint-angle:

```
(def-assemblage-maker joint-angle (name)
  ((output-position 0 -45 45)
   (sensor-position 0 -45 45)
   (destination 0 -45 45)
    :slots flag)
```

Lift-lower assemblage : this controls the lift joints of the four legs.

```
(def-assemblage-maker lift-lower (name leg-name lower-leg-name)

 ((output-port 0 -45 45)
  (input-port-sensor 0 -45 45)
  (input-port-walk 0 -45 45))
```

There are four assemblage instances created corresponding to the four legs

```
(lift-lower 'lift-lfl lfl lower-lfl)
(lift-lower 'lift-lfr lfr lower-lfr)
(lift-lower 'lift-lrl lrl lower-lrl)
(lift-lower 'lift-lrr lrr lower-lrr)
```

It is noticed that every time an instance is created there are three arguments , the name of the lift-leg instance, the name of the leg and the name of the lower-leg instance.

Swing-leg assemblage: this is similar to the lift-lower assemblage , it is responsible for controlling the swing/stance of each leg.

Stick-assemblage: this assemblage is always active commanding the sucker mechanism to stick to the wall unless a suppress message is generated by the Move-assemblage instance.

3.3 The Gait Controller Layer

The gait controller layer consists of two behaviors on the top of the locomotion layer Fig. 2.

Move-assemblage : There is only one assemblage instance generated, it is used to sequence and coordinate the movements of the robot legs. Using the arbitration mechanism (suppression and inhibition nodes) several gaits could be implemented for walking and climbing .

Balance-Assemblage: A single assemblage instance is generated every time this procedure is called.

The balance machine is used during the robot movement only. It compensates each leg swing movement as to maintain the stability of the robot.

4 CONCLUSION

A total of 28 assemblage instances (behaviors) have been designed under the locomotion layer. On top of the locomotion layer, two other behaviors are under development to implement different types of gaits.

Our approach in designing the control architecture of a wall climbing Robot is based on distributing the control algorithm. This is achieved by using the subsumption paradigm which enables us to implement a multiprocessing architecture.

Using the L development package we have been able to develop the complete subsumption code on a host computer and then download it to the master processor.

REFERENCES

[1] Rodney A. Brooks, A robust layered control system for mobile robot, A.I. Memo 864, Artificial Intelligence Laboratory, MIT, May 1986.

[2] Rodney A. Brooks , "A layered Intelligent Control System for a Mobile Robot", IEEE Journal Robotics And Automation . RA-2, April, 14-23.

[3] Rodney A. Brooks , " Achieving Artificial Intelligence Through Building Robots ", A.I.Memo 899, Artificial Intelligence Laboratory, MIT, May 1986.

[4] Rodney A. Brooks , " A robot that walks ; Emergent behaviors from a carefully Evolved network", Proceedings of the International Conference on Robotics and Automation 692-4 Vol. 2 , May 1989 . (Also A.I. Memo 1091 , Artificial Intelligence Laboratory, MIT).

[5] Rodney A. Brooks , " The world's Largest One cubic Inch Robot", Proceeding of the IEEE Micro-Electromechanical Systems, Feb. 1989.

[6] Rodney A. Brooks, Anita M. Flynn , William M. WellsIII, David S. Barrett , " Squirt: The Prototypical Mobile Robot For Autonomous Graduate Students", A.I. Memo 1120, Artificial Intelligence Laboratory, MIT

[7] Rodney A. Brooks and Anita M. Flynn, " Battling Reality ", A.I. Memo 1148 , Artificial Intelligence Laboratory, MIT.

[8] Rodney A. Brooks , " The Behavior Language ; User's Guide ", A.I. Memo 1227, Artificial Intelligence Laboratory , MIT.

[9] Rodney A. Brooks , " Intelligence without Reason ", Computers and thought, IJCAI-91, Also A.I. Memo 1293 Artificial Intelligence Laboratory, MIT.

[10] Rodney A. Brooks , MARS: Multiple Agency Reactivity System, (MARS Manual) Version of January 29, 1996.

[11] Cynthia Ferrell, " Robust Agent Control Of an Autonomous Robot With Many Sensors and Actuators", Msc Thesis, Artificial Intelligence Laboratory, MIT, (USA).

[12] Cynthia Ferrell " Global Behavior Via cooperative Local Control", Artificial Intelligence Laboratory, , MIT Internal Report.

[13] Cynthia Ferrell, "Failure Recognition and Fault Tolerance of an Autonomous Robot" , Artificial Intelligence Laboratory, MIT Internal Report.

[14] Cynthia Ferrell, " A comparison of three Insect-inspired Locomotion Controllers", Robotics and Autonomous Systems, ELSEVIER, 16 (1995) 135-159.

[15] Jonathan H. Connell, " A colony Architecture for an Artificial Creature" , Technical Report 1151, Artificial Intelligence Laboratory, MIT.

[16] Jonathan H. Connell , " A Behavior Based Arm Controller", IEEE Journal of Robotics And Automation, (Also A.I. Memo 1025 , Artificial Intelligence Laboratory, MIT).

[17] Maja J. Mataic, " Interaction and Intelligent Behavior", Ph.D. Thesis , Department of Electrical Engineering And Computer Science, Artificial Intelligence Laboratory, MIT.

[18] Ronald C. Arkin , "Motor Schema Based Navigation for a Mobile Robot", Proceeding of the international conference on robotics and automation , PP 264-271, April 1987.

[19] Ronald C. Arkin , " The Impact of Cybernetics on the Design of a Mobile Robot System: A case study", IEEE Transactions on Systems, Man , and Cybernetics Vol. 20 No 6 Nov./DEC 1990.

[20] Ronald C. Arkin, " Reactive control as a Substrate for Telerobotic Systems", IEEE AES Systems Magazine, June 1991.

[21] Shinichi Nakasuka, Takehisa Yairi, Hiroyuki Wajima, " Autonomous Generation of Reflection-based Robot Controller using Inductive Learning" Robotics and autonomous Systems , ELSEVIER, 17(1996)287-305.

[22] Kenneth Moorman and Ashwin Ram , " A Case-Based Approach to Reactive Control for Autonomous Robots", AAAI Fall Symposium " AI for real time Mobile Robots", Cambridge MA, Cot 1992.

[23] Sunil Cherian , Wade O. Troxel, and Muhammad M. Ali. " Design Of Behavior-Based Micro Rover Robot".

[24] James S. Albus, "Hierarchical Control Of Intelligent Machines Applied to Space Station Telerobots" IEEE Transactions on aerospace and electronic systems, Vol. 24. NO. 5 Sept. 1988.

[25] James S. Albus "Overview of the Multiple Autonomous Underwater Vehicles (MAUV) project" , IEEE, 1988.

[26] Erann Gat "ALFA: A language for Programming For Action" proceeding of the 1991 IEEE International Conference on Robotics and Automation Sacramento, California-April 1991.

[27] Erann gat, Behavior Control for Planetary Exploration: Interim Report , Rajiv, Robert Ivlev, John Loch, and David P. Miller.

[28] Erann gat , Rajiv, Robert Ivlev, John Loch, and David P. Miller "Behavior Control for Robotic Exploration of Planetary Surfaces", IEEE Transactions on Robotics and Automation, Vol. 10,NO4, August 1994.

[29] Erann gat The ESL user's guide Jet Propoltion Laboratory, Erran gat 12 August 1996.

[30] R. James Firby The CRL Manual , University of Chicago Animate Agent Working Note AAP-3, version 1.1 , February 1995.

[31] R. James Firby The RAP Language Manual , University of Chicago Animate Agent Working Note AAP-6, version 1 , March 1995.

[32] Housheng Hu and Michael Brady, " A Parallel processing architecture for sensor based control of intelligent mobile robots" Robotics and Autonomous Systems, PP, 235-257 ,17 ,1996.

[33] Nadir Ould Khessal , Shamsudin H. M. Amin "Behavior-Based Control Architecture for a Wall Climbing Robot", IEEE International Conference on Intelligent Engineering Systems, September 1997, Budapest, Hungary.

[34] Nadir Ould Khessal , Shamsudin H. M. Amin "Behavior-Based Artificial Intelligence and Robotics", Proceeding of Research Seminar on 'Electrical, Electronics, Aerospace Information Technology and Communications' , 7 October 1997 Sofitel Hotel, Senai, Johor, Malaysia.