A Visual Control Architecture for a Virtual Humanoid

Nathan Sprague and Dana Ballard

Computer Science Department
University of Rochester
Rochester NY 14627
{sprague,ballard}@cs.rochester.edu

Abstract

We present a visual control architecture for a virtual humanoid agent that inhabits a realistic simulated world. The goal is to develop a vision system that is flexible enough to handle a wide range of tasks in a complicated and dynamic 3D environment. Vision in this system is procedural: when some piece of information is required for a task, a special purpose visual routine is executed to gather that information.

Visual routines are employed by *visual behaviors* that each handle a single well defined visuo-motor task, such as catching a ball or avoiding an obstacle. The behavior-based approach was developed in the robotics community, and here we extend it in several respects. In particular, there has been no explicit recognition that behaviors must compete for limited computational resources in addition to the resources of the body. We propose a solution to this difficulty that draws on ideas from the area of computer operating systems.

1. Introduction

In the past, combining vision with action has required robot hardware that is generally expensive, difficult to maintain, and severely limited in its capabilities. The advent of powerful 3D graphics hardware has enabled an alternative to physical robots. Terzopoulos and Rabie have pioneered the idea of employing simulated agents in virtual environments as a basis for active vision research. In this approach, visual processing is performed on a rendered video stream rather than the output of digital cameras. Terzopoulos' original virtual agents were fish that used color information and vergence to chase prey and avoid predators [11].

1.1 Virtual Humanoids

Recently, Rabie and Terzopoulos have transplanted the visual capabilities of their virtual fish to a virtual humanoid agent [8]. Virtual humanoids represent a promising direction for vision research. One advantage of virtual humanoids is the wealth of data available on how people deal with visual tasks. This information provides a baseline for evaluating the performance of virtual agents. Human data can also provide hints about how a successful vision system should be organized.

Another important benefit of using a virtual humanoid, rather than another type of virtual agent, is related to the issue of embodiment. Human intelligence is optimized to take advantage of the particular form of the human body [1]. More generally, it is difficult to conceive of intelligence without any sort of physical embodiment [3]. This leads to the conclusion that the particular physical characteristics of an agent must be considered when it comes to imbuing that agent with intelligence. Since we are primarily interested in human-like intelligence, it makes sense to use a simulated agent that has a human form.

1.2 Humanoid Vision

In our research, we use virtual humanoids as a platform for studying methods of organizing visual processing. Basic computer vision research has made great progress in recent years. However, there is still little understanding of how to effectively utilize visual information to subserve the goals of mobile agents in dynamic environments. The central difficulty is that there is no general purpose computer vision algorithm; successful computer vision systems invariably tackle a single well defined problem. The question then becomes: how is it possible to achieve behavioral generality, when computational feasibility demands the use of special purpose vision algorithms?

Ullman's *visual routines* paradigm [12] provides a possible answer to this question. The visual routines approach suggests that visual processing is built on top of a library of simple visual operators. Generality is achieved by composing multiple visual operators into task specific visual routines. The crucial property of the visual routines approach is that vision is procedural; when the performance of some task requires a piece of information, a visual routine is run that collects that information. This is an efficient approach because only the necessary visual processing is performed. There is also compelling evidence that human vision has this procedural character. For a review of the relevant human research refer to [5].

The challenge in designing a system based on visual routines lies in scheduling the routines to support ongoing behavior. In part, the proposed approach is related to research in behavior based robotics. The be-

Proceedings of the IEEE-RAS International Conference on Humanoid Robots
Copyright @ 2001

Figure 1: An image of the virtual humanoid in his environment.

havior based approach eschews centralized control in favor of many separate control modules, each in charge of a single well defined task. Similarly, we organize visual routines into *visual behaviors*. Each visual behavior employs visual routines to accomplish a single narrowly defined goal. An example of a visual behavior that will be discussed later is sidewalk following. The sidewalk following behavior uses a visual routine to detect the edge of the sidewalk, and reactively steers the agent to maintain a constant distance from it.

The unspoken assumption in behavior based control has been that there is sufficient processing power for all necessary behaviors to run in parallel. The simultaneously active set of behaviors each suggests some action, and the focus is on arbitrating between the suggested actions to choose the best one. The assumption of unlimited processing power is appropriate when behaviors are simple reactive mappings from a low dimensional sensor space to a low dimensional action space. However, it is less appropriate when behaviors are required to perform expensive image processing.

The approach taken in this paper is to restate behavior based control as a resource allocation problem. In particular, we consider individual behaviors to be analogous to threads of execution in a computer operating system. Behaviors, like computer processes, compete for a finite amount of processor time and for control of other limited resources. In the architecture presented in this paper, this competition will take the form of a set of auctions, where each behavior places bids for the resources it needs.

2. Simulation Platform

The virtual humanoid simulation environment is composed of several different software components. Wherever possible we have relied on off the shelf software packages for quick development and ease of use.

The rendering is handled by SGI's Performer package. The visual environment is a highly detailed 3D model of a small town including buildings, roads, and vehicles.

In order to provide our virtual world with physics, we use the Vortex simulation package from Critical Mass Labs. This package provides stable and fast kinematic simulation and contact detection.

The graphical humanoid agent itself is built on Boston Dynamic's DI-Guy. DI-Guy provides visually realistic and highly efficient human motion generation based on motion capture. The disadvantage of motion capture based animation is that characters are limited to a library of pre-recorded motions. We have overcome this limitation by constructing an articulated arm driven by simple endpoint control. The arm is built on top of Vortex and is thus able to physically interact with the environment.

Figure 1 shows the virtual humanoid. Although this image shows the humanoid juggling using his physics based arm, the work presented in this paper will not make use of the arm, or the physical simulation software.

2.1 Simulated Vision

There are well recognized dangers associated with working in a simulated environment. No simulation can be an entirely accurate and complete representation of the real world. It is always necessary to decide which variables are important to represent, and which are irrelevant. If one is careless in making these decisions, it is possible to create a solution that works in simulation, but is not relevant to solving any real world task.

There are many such pitfalls in developing vision algorithms that process computer rendered video. Even though a rendered image may look convincing to the human eye, the pixel level characteristics will generally be very different from real images. This limits the type of research questions that can be reasonably addressed in simulation. This is not a serious problem for the work presented in this paper, because our focus is not on the details of image processing, but on higher level issues of visual control. Given this, we are willing to settle for qualitatively realistic vision: The image processing performed by the simulated agent is not required to be transferable as-is to a real world domain. However, it is should be within the realm of what can be accomplished using real image data.

3. Visual Architecture

The vision architecture that we develop borrows from ideas developed in the robotic agents community [10, 2, 6]. In particular, the small autonomous robots developed for Robo-Cup have incorporated ideas of re-

Copyright @ 2001

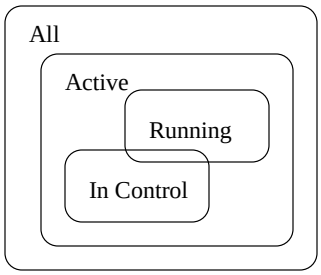

Figure 2: The organization of behaviors.

source competition and state-dependent control of actuators. We extend these ideas to two levels of competition. Visual routines compete for resources in order to update their measurements without actually gaining control of the body. A separate round of competition is used to determine which of the running routines will control the body.

Figure 2 demonstrates the basic organization of behaviors. The outermost circle contains all of the behaviors in the agent's repertoire, most of which will be irrelevant at any given time. The middle circle, labeled "active", contains all of the behaviors that are appropriate for controlling the agent given the current task goals and environmental conditions. The circle labeled "running" contains the behavior(s) that are currently executing. The fact that this is a subset of all of the relevant behaviors is a consequence of the fact that the agent has limited computational resources. In other words, it is possible that there are more tasks vying for the agent's attention than can be handled simultaneously. The final circle, labeled "in control", contains the behaviors that are currently controlling the agent's body.

Behaviors must move between the regions in Figure 2 in order for the agent to accomplish his long term goals. Each of these transitions represents a different task that must be handled by the architecture. The main focus of this paper is on the two innermost circles: selecting behaviors to run, and determining which behaviors should be given control of the body. The problem of selecting a set of relevant active behaviors is left as future work.

3.1 Scheduling

The set of relevant behaviors will generally evolve over the course of seconds or minutes as the agent's overall situation changes. However, there is also a need for a scheduling mechanism to select behaviors to execute at a finer time scale. This corresponds to moving a behavior from "Active" to "Running" in Figure 2. Each of the agent's active behaviors is responsible for tracking some aspect of the environment and responding to it appropriately. Depending on the nature of the information that is being tracked, a given behavior may need to run more or less frequently.

The goal of the scheduling mechanism is to ensure that each active behavior gets a large enough share of the processor to meet its own goals. More specifically the scheduler chooses a single behavior at every video frame and allows that behavior to execute. The frame rate of the simulation is approximately 12HZ, so the scheduler is run about 12 times per second. When a behavior is chosen to run, it is allowed to move the eyes, and it is given exclusive access to the agent's visual input.

There is nothing special about these particular numbers. The issues are the same if more than one behavior executes in parallel, as long as the total number that runs at once remains small relative to the total number of relevant behaviors. The issues are also be the same if the size of the quanta is longer.

When designing an operating system there are a wide range of possible scheduling mechanisms to choose from, from simple round-robin selection to elaborate schemes with real time guarantees. The scheduler in this paper is based on the lottery scheduling algorithm [13]. Lottery scheduling is a simple, but elegant mechanism that allows proportional share resource distribution, and avoids starving processes. It works as follows: Each process is allocated a certain number of tickets. The scheduler randomly selects one ticket from the entire set of tickets in existence. The process that holds the winning ticket is then allowed to run. Under this scheme, the probability that a process will run is directly proportional to the number of tickets that it holds.

The challenge, from the perspective of the visual architecture, is determining how many tickets each behavior should have at any given time. One alternative is to assign a fixed number of tickets to each behavior, so that each receives an appropriate share of the processor. However, it should be possible to achieve better performance by dynamically re-prioritizing the behaviors as the agent's situation changes.

We implement such a dynamic prioritization mechanism by allowing behaviors to bid for cycles based on their perceived need. At each video frame all behaviors place a bid in the range 0–1 indicating what share of the processor they would like to have. The architecture analyzes all of these bids, and allocates tickets appropriately. Then the scheduler is run, the winning behavior is given an opportunity to update its bid, and the process starts over. At present the architecture allocates tickets in direct proportion to the behaviors relative bid; if one behavior bids twice as high as another, it will receive twice as many tickets.

Copyright @ 2001

3.2 Arbitration

If each behavior is allowed to take control of the body every time it is executed by the scheduler, then the agent will be susceptible to dithering; control of the body may rapidly alternate between multiple behaviors in such a way that no behavior is able to accomplish its goal. In order to avoid this, control of physical resources is allocated separately from processing cycles. At each frame behaviors place bids for control of the body. The behavior that bids highest is given control. In addition to reducing dithering, this mechanism allows separate allocation of cycles and bodily resources; it may not always be the case that the behavior that needs the most cycles also has the greatest need to control the body.

Putting a behavior in control of the body is equivalent to moving it from "Running" to "In Control" in Figure 2. "In Control" is not a proper subset of "Running" because a behavior maintains control of the body as long as its bid is highest, even if it is not actively gathering data from the environment.

4. Sidewalk Navigation

The visual architecture outlined above is demonstrated on a simple sidewalk navigation task. In this task the agent must walk along a sidewalk while simultaneously meeting two goals: avoiding a number of moving obstacles, and maintaining a constant distance from the street. Each of these goals is maintained by an independent visual behavior. Despite the simplicity of this task it contains many of the challenging aspects of vision, including time constraints, and conflicting priorities.

4.1 Sidewalk Following

The sidewalk following behavior proceeds in the following steps: First, a border operator is applied to label pixels on the border between the sidewalk and the street. Next, a line detection operator is employed to find the most prominent line in the border image. Finally, rays are cast into the image to retrieve the 3D position of two points on the line in order to compute a 3D vector that indicates the absolute position and direction of the sidewalk in space.[1] The recovered coordinates are quite noisy, so some simple smoothing is performed. The agent's current distance from the street is estimated, and a simple reactive algorithm is used to steer him away from the street if he is too close, or toward the street if he is too far away. The steps of the visual processing for this behavior are summarized in Figure 3.

[1] If the use of absolute position information seems unreasonable, keep in mind that all of this could be computed relative to the virtual humanoid. In fact it is not necessary to use the depth operator at all. It would be possible to estimate the relative 3D sidewalk direction directly from image coordinates.

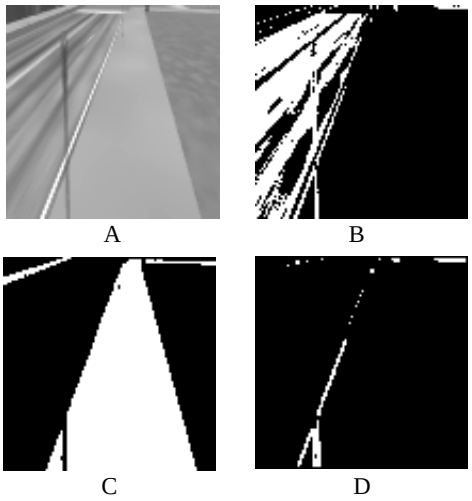

Figure 3: The steps of visual processing for the sidewalk following behavior. **A** shows the unprocessed visual input that the agent receives. He is looking down at the sidewalk ahead. **B** shows the parts of the image that match the color profile of the street. **C** shows the parts of the image that match the color profile of the sidewalk. The results here are much cleaner than in **B** because the sidewalk is distinctly and uniformly colored. **D** shows the pixels that are on the border of the street and the sidewalk. The final step of the process is matching a line to these pixels.

Recall that behaviors bid for resources according to their perceived needs. The two resources in question here are processing time and control of the agent's legs. Parts A and B of Figure 4 summarize the bidding strategy of the sidewalk following behavior. Part A illustrates that the leg bid is a monotonically increasing function of the agent's perceived distance from his ideal location on the sidewalk. As can be seen from the figure, the minimum leg bid is a small non-zero value. This ensures that in the absence of any competing behaviors, the sidewalk following routine will remain in control of the agent's direction at all times. The exact shape of this bid function, and the others that we will discuss, are not derived in a principled way. Instead, they were designed by hand, and tuned slightly to give reasonable performance. Future work will focus on less labor intensive approaches to determining appropriate bidding strategies.

Part B of Figure 4 illustrates the strategy that the sidewalk following routine uses to bid for cycles. In this case the bid value is a function of a running estimate of the standard deviation in the perceived direction of the sidewalk over the last several frames. When there is little variability in the perceived direction, the

Copyright @ 2001

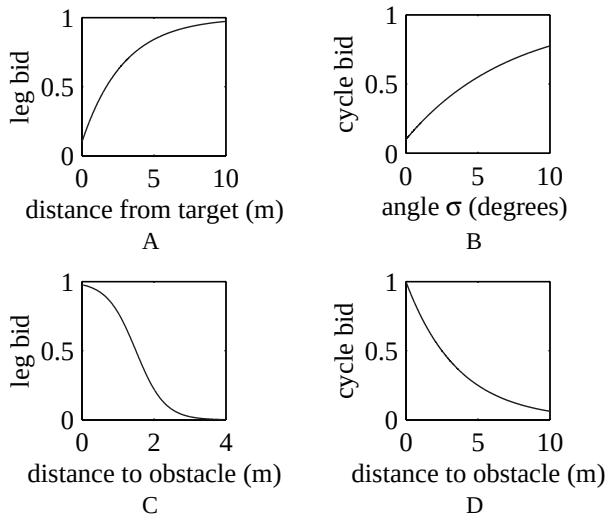

Figure 4: The bidding strategies of the sidewalk following behavior and the obstacle avoidance behavior. The Y axes represent real valued bids in the range 0-1. In all but **B** the X axis is in meters. In **B** the X axis is the standard deviation of the perceived sidewalk angle over the last few frames.

assumption is that the current estimate of sidewalk direction is accurate, and few cycles need to be used to refine that estimate. On the other hand, when variability is high, the assumption is that either the direction is changing, or the agent is receiving noisy readings due to occlusion or other distractions in the environment. In either case it makes sense to dedicate more cycles to determining the true sidewalk direction.

4.2 Obstacle Avoidance

The obstacle avoidance behavior does not operate directly on the image pixels. instead it simply casts a number of rays into the world model to obtain a rough depth map of the area ahead. The rays fan out from the agent's eyes in a one dimensional horizontal plane covering the agent's field of view. If any of these rays indicate that an obstacle is nearby, the obstacle avoidance routine will bid for control of the legs. It suggests a direction that will clear the nearest obstacle in the agent's path with the minimum deviation from the current direction. If there are no obstacles in the agent's path, the behavior maintains the current direction.

This behavior will request control of the agent whenever there are obstacles nearby, even if the agent is not currently on a collision course with those obstacles. This is necessary because the behaviors do not communicate–there is no way for the obstacle avoidance behavior to know that some other behavior will not steer directly into a nearby obstacle if given control.

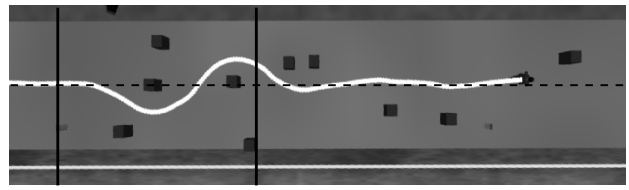

Figure 5: An overhead view of the virtual humanoid performing the sidewalk navigation task. The curving white line indicates the agent's path. the dotted horizontal line indicates the preferred path for the sidewalk following behavior. The agent finds a reasonable compromise between staying on the preferred path and avoiding obstacles. The dark vertical lines indicate the portion of the agent's path that will be analyzed more closely in the following figures. In this image, the obstacles remain stationary.

Parts C and D of Figure 4 illustrate the bidding strategy of the obstacle avoidance routine. Both bidding for control of the legs, and bidding for cycles are based on the distance to the nearest obstacle. As the agent gets closer to an obstacle it becomes more important to initiate a turn, and more cycles are required to track the relative location of the obstacle.

4.3 Sidewalk Navigation Example

Figure 5 shows an example of the virtual humanoid performing the sidewalk navigation task. In the interest of clarity the obstacles in this example remain stationary. The agent successfully avoids all obstacles, while staying close to the desired path. In the next few figures we examine the interaction of the two behaviors in some detail. The time period examined is indicated by the two vertical black lines in Figure 5.

Figure 6 shows the frame by frame leg bids placed by the two behaviors during the time period under consideration. Initially there are no obstacles nearby, and the sidewalk following routine has control of the agent. As the agent approaches each obstacle, the obstacle avoidance routine begins to ramp up its bid for control of the legs until it eventually takes control.

Each time the obstacle avoidance behavior takes over, the agent is pulled away from the path preferred by the sidewalk following behavior. The sidewalk following behavior responds by increasing its own bid.

Figures 7 and 8 demonstrate the process of bidding for cycles in the sidewalk navigation task. Both figures cover exactly the same time period as Figure 6. Figure 7 shows the changing bids for cycles over the time period. The sidewalk bid remains fairly constant, while the bid placed by the obstacle avoidance behavior in-

Copyright @ 2001

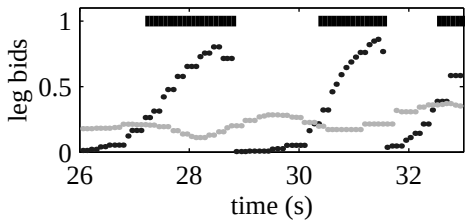

Figure 6: Bidding for control of the legs. This figure shows a frame by frame trace of the bids placed for control of the legs by the two behaviors. The black points are the obstacle avoidance bids, and the gray points are the sidewalk following bids. The black bars at the top of the figure indicate the time periods during which the obstacle avoidance behavior has taken control by placing the highest bid. Refer to Figure 5 to see the agent's behavior during this time period.

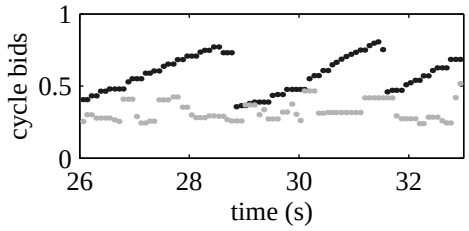

Figure 7: Bidding for cycles. This figure shows a frame by frame trace of the two behavior's bids for processing cycles. The black points are the obstacle avoidance bids, and the gray points are the sidewalk following bids. The obstacle avoidance behavior requests a higher proportion of cycles when it detects obstacles nearby.

creases as the agent approaches each obstacle. Figure 8 shows the resulting priorities and indicates which behavior actually executes on each frame. The effect of changing priorities can be seen clearly between seconds 29 and 31. In the period from second 29 to second 30, the priorities for each behavior are nearly equal. During this time period both behaviors execute with nearly equal frequency. In contrast, during the time period from second 30 to second 31, the agent is near an obstacle, and the priority for obstacle avoidance increases. The obstacle avoidance behavior is granted the majority of cycles in this period.

5. Results

In order to test the effectiveness of the virtual humanoid's visual architecture on this task we set up a fixed obstacle course, and recorded the agent's success as he repeatedly navigated a short stretch of sidewalk.

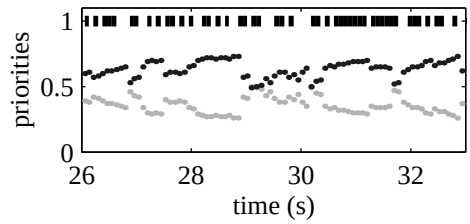

Figure 8: Changing priorities. This figure shows the changing priorities of the two behaviors as the proportion of the number of lottery tickets that each holds. The points in this figure are derived from those in Figure 7 simply by rescaling the bids so that they sum to one. They can be interpreted as the probability that each of the two behaviors will be executed during a given frame. The black tick marks along the top of the figure indicate the frames where the obstacle avoidance routine is executed. In frames lacking a tick mark, the sidewalk following routine is executed. The black points are the priorities for obstacle avoidance, the gray points are for sidewalk following.

The obstacles are a set of 12 rectangular solids, four meters tall and forty centimeters on a side. The initial placement of these obstacles was determined randomly, and remains constant over all trials. A portion of the course can be seen in Figure 5. In order to make obstacle avoidance more difficult, the objects move perpendicularly to the direction of the sidewalk in a sinusoidal path with an amplitude of .6 meters, and a period of 6 seconds. For this experiment no collision detection is performed. If the agent hits an obstacle he simply passes through it.

In the current task, the agent is only required to get to the end of the sidewalk without colliding with anything. However, it is easy to imagine a situation where the sidewalk navigation behaviors must coexist with other goals that vie for the agent's attention. For example, it may be necessary to look out for a taxi, avoid stepping in puddles, or do some window shopping. Each of these behaviors may steal cycles that could otherwise be used for sidewalk navigation. In order to simulate this effect without coding a large number of additional behaviors, we introduce a NO-OP behavior that competes for cycles but performs no useful work. In the next two figures we will examine the agent's success on the sidewalk navigation task as a function of the percentage of cycles that are consumed by the NO-OP process.

One premise of the humanoid's visual control architecture is that it is possible to make more efficient use of resources by dynamically re-weighting priori-

Copyright @ 2001

ties as the agent's situation changes. In order to test this claim, the bid-based dynamic scheduling mechanism described above is compared to a static scheduling mechanism where each behavior is given a fixed priority. The lottery scheduling mechanism is still used, but each behavior maintains a fixed number of tickets over the entire trial. In the fixed condition, the sidewalk following behavior is given 40% as many tickets as the obstacle avoidance behavior. This ratio was chosen by setting the NO-OP rate to 50% and systematically varying the relative priorities to find the best compromise. The two conditions differ only in the method used to allocate cycles. Control of the body is handled in exactly the same way in all cases.

Figures 9 and 10 show the results of the sidewalk navigation task for both the dynamic and static scheduling schemes. Each data point is obtained by taking the mean of 20 trials. Each trial takes approximately 25 seconds.

Figure 9 shows the performance of the sidewalk navigation behavior. Performance is measured as the average deviation from the target path over the course of the trial. It can be seen that the dynamic scheduling mechanism performs much better than the static mechanism in the face of increasing load. The performance stays fairly constant for both until the number of NO-OP cycles reaches 50%. At that point the performance of the dynamic scheduling mechanism begins to degrade gradually, while the static scheduling mechanism begins to suffer dramatically. The 90% NO-OP data point is not present for the static condition, because the agent was unable to consistently reach the end of the course at that load.

Figure 10 examines how well the agent is able to avoid collisions. Rather than counting the number of collisions, we record the amount of time that the agent spends in contact with obstacles. This provides a more sensitive measure; a glancing blow will be less costly than walking directly through an obstacle. The performance of the static and dynamic scheduling mechanism are fairly similar in this figure. The performance of both degrades slowly as the number of NO-OP cycles approaches 40%-50%, and then degrades more rapidly from that point on. The static scheduling mechanism seems to perform slightly better overall, but the difference is not dramatic.

One possible factor in the relatively good performance of the static scheduling mechanism is that fact that all obstacles are located on the sidewalk. As can be seen in Figure 9, by the time the NO-OP rate reaches 60% the agent is spending a fair amount of time off of the sidewalk entirely in the static condition. Thus the failure of the sidewalk following behavior may be indirectly benefiting the obstacle avoidance behavior by keeping the agent in the street or on the lawn where there are no obstacles.

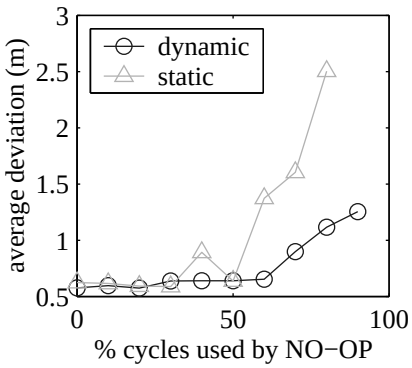

Figure 9: Performance of the sidewalk following behavior as a function of load. In the case of the dynamic scheduling mechanism the priorities for the two behaviors are re-weighted over the course of each trial in response to changing bids. For the static scheduling mechanism each behavior has a fixed priority. The fixed priority mechanism does not perform as well as the load increases.

6. Related Work

An early example of studying vision through simulation is provided by the work of Chapman. Chapman [4] demonstrated visual routines in a two dimensional video game which featured an agent SONJA. These visual routines were simulated in the sense that they accessed the world model directly without performing any real image processing.

Salgian [9] demonstrated, in a virtual environment, that many aspects of driving, including responding to traffic signs and car following, can be handled by using a simple scheduling protocol to run special purpose visual behaviors that each handle a single aspect of the problem. These are applied in a fixed sequence that ensures that each is called at a rate that is appropriate for the control problem that it is addressing. Salgian's work focused more on the details of vision, and less on the architectural organization than the work presented here.

There is a large body of work in behavior based robotics that is relevant to the research presented here. In particular, there has been a good deal of work in developing behavior coordination mechanisms. A good review of that research can be found in [7].

In his thesis work Robert Wisniewski [14] proposes operating system mechanisms for supporting artificial intelligence applications. He emphasizes that the architecture must be able to adjust to the ongoing demands of the application. Although his system is

Copyright @ 2001

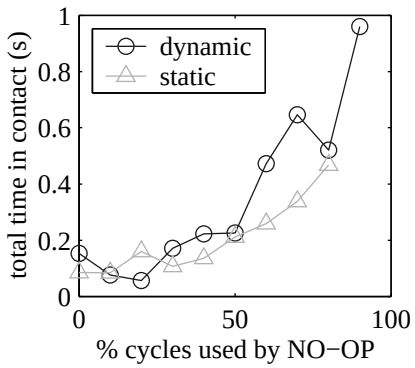

Figure 10: Performance of the obstacle avoidance behavior as a function of load. Here the performance of the dynamic and static scheduling mechanisms are similar.

demonstrated on a very simple simulated sheep herding task, his architecture has many similarities to the work presented here.

7. Discussion and Future Work

The work presented in this paper makes contributions on two fronts. First, it demonstrates the usefulness of virtual humanoids as a testbed for vision research. Second, it shows the promise of a finer grained view of resource allocation in behavior based control. In order to achieve optimal performance it must be recognized that behaviors not only compete for control of the body, but also for computational resources and access to sensors. The sidewalk navigation task demonstrates that it is possible to improve performance by focusing computational resources where they are needed most, independently of which behavior currently has control of the body.

The main disadvantage of the visual architecture presented here is the requirement that the voting strategies of the behaviors be hand tuned for optimal performance. In the case of only a few behaviors this is not a difficult task. However, it is likely to become a design bottleneck as the number of behaviors is scaled up. In order to address this, future work will focus on machine learning approaches both for learning appropriate bidding strategies at the level of behaviors, and learning to properly weight bids at the level of the architecture.

Acknowledements

This material is based upon work supported by the National Institutes of Health Public Health Service research grant number 5 P41 RR09283. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necesarily reflect the views of NIH.

References

[1] Dana Ballard, Mary Hayhoe, and Polly Pook. Deictic codes for the embodiment of cognition. *Behavioral and Brain Sciences*, 20:723–767, 1997.

[2] Randall D. Beer. The dynamics of adaptive behavior: a research program. *Robotics and Autonomous Systems*, 20:257–289, 1997.

[3] Rodney A. Brooks. Intelligence without reason. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, pages 569–595, Sydney, Australia, 1991.

[4] David Chapman. *Vision, Instruction and Action*. PhD thesis, MIT AI Lab, 1990.

[5] Mary Hayhoe. Vision using routines: a functional account of vision. *Visual Cognition*, 7:43–64, 2000.

[6] Herbert Jaeger and Thomas Christaller. Dual dynamics: Designing behavior systems for autonomous robots. *Artificial Life and Robotics 2*, pages 108–112, 1998.

[7] Paolo Pirjanian. Behavior coordination mechanisms – state-of-the-art. Technical report, Institute for Robotics and Intelligent Systems, University of Southern California, October 1999.

[8] Tamer F. Rabie and Demetri Terzopoulos. Active perception in virtual humans. In *Proc. of the thirteenth Canadian Vision Interface Conference (VI '2000)*, Montreal, Quebec, May 2000.

[9] Garbis Salgian. *Tactical Driving using Visual Routines*. PhD thesis, University of Rochester, Computer Science Dept., December 1998.

[10] Luc Steels. Building agents out of autonomous behavior systems. In L. Steels and R.A. Brooks, editors, *The "Artificial Life" Route to "Artificial Intelligence": Building Situated Embodied Agents*. Lawrence Erlbaum, 1993.

[11] Demetri Terzopoulos and Tamer F. Rabie. Animat vision: Active vision in artificial animals. *Videre: Journal of Computer Vision Research*, 1(1):2–19, 1997.

[12] Shimon Ullman. Visual routines. *Cognition*, 18:97–159, 1985.

[13] Carl A. Waldspurger and William E. Weihl. Lottery scheduling: Flexible proportional-share resource management. In *Proc. Symp. on Operating Systems Design and Implementation*, pages 1–12, Monterey CA (USA), 1994.

[14] Robert Wisniewski. *Achieving High Performance in Parallel Applications via Kernel-Application Interaction*. PhD thesis, University of Rochester, Computer Science Dept., April 1996.

Copyright @ 2001