From: AAAI-92 Proceedings. Copyright ©1992, AAAI (www.aaai.org). All rights reserved.

A Reactive Robot System for Find and Fetch Tasks in an Outdoor Environment

R. Peter Bonasso, H. James Antonisse, Marc G. Slack

Autonomous Systems Laboratory, the MITRE Corporation
7525 Colshire Drive
Mclean, Virginia 22102
pbonasso@mitre.org

Abstract

This paper describes the results of using a reactive control software architecture for a mobile robot retrieval task in an outdoor environment. The software architecture draws from the ideas of universal plans and subsumption's layered control, producing reaction plans that exploit low-level competences as operators. The retrieval task requires the robot to locate and navigate to a donor agent, receive an object from the donor, and return. The implementation employs the concept of navigation templates (NaTs) to construct and update an obstacle space from which navigation plans are developed and continually revised. Selective perception is employed among an infrared beacon detector which determines the bearing to the donor, a real-time stereo vision system which obtains the range, and ultrasonic sensors which monitor for obstacles en route. The perception routines achieve a robust, controlled switching among sensor modes as defined by the reaction plan of the robot. In demonstration runs in an outdoor parking lot, the robot located the donor object while avoiding obstacles and executed the retrieval task among a variety of moving and stationary objects, including moving cars, without stopping its traversal motion. The architecture was previously reported to be effective for simple navigation and pick and place tasks using ultrasonics. Thus, the results reported herein indicate that the architecture will scale well to more complex tasks using a variety of sensors.

Introduction

We are interested in programming robots to carry out tasks robustly in environments where events for which the robot has a response can occur unpredictably, and wherein the locations of objects and other agents are usually not known with certainty until the robot is carrying out the required task. To achieve this we have taken the situated reasoning approach which closely integrates planning and plan monitoring through perception in real-time. The approach requires that the planning system selectively use the various sensors available to the robot to track changes in the environment and to verify that plan tasks are achieved.

We are using an agent software architecture wherein reaction plans (Schoppers 1989) use subsumption competences (Brooks 1986) as operators. The benefits are that reaction plans need only map agent states into competences, thus reducing their size, and that formal goal representations may be used to augment the limited representational power of the subsumption competences.

Our architecture is implemented in the GAPPS/Rex programming language (Kaelbling 1988), which expresses and enforces a formal semantics between an agent's internal states and those of the environment (Rosenschein and Kaelbling 1986). The reaction plans and the layered competences are expressed in propositional reduction rules, augmented by Common LISP programming constructs. They are subsequently compiled into a circuit formalism which supports direct analysis of parallelism and enforces a constant time cycle execution regimen. Additionally, the accompanying programming environment supports generating the circuit code in C and other languages in which commercial robot control operating systems are written.

Our previous work (Bonasso 1991a, Bonasso 1991b) showed the efficacy of our agent architecture for following a trajectory among obstacles on land and underwater, and for pick and place tasks. Though these tasks were useful, particularly with human supervision, they required only ultrasonic and directional sensing and were of relatively low complexity. We desired to see how the architecture would scale to more complex tasks with more sophisticated sensors. The find and fetch task described below is both useful for robots to carry out and of sufficient complexity to suggest a positive answer to the scaling question.

The Find and Fetch Task

Our land mobile robot find and fetch task requires a robot to locate a donor agent which holds an object to be retrieved, to navigate safely among obstacles to the donor, to accept the object from the donor, and to return to its original location. Our goal was to generate a reaction plan for one of our mobile robots to carry out such a task outdoors in an area which held an unpredictable number of stationary and moving objects. A parking lot provided such an environment, due to the intermittent appearance of moving cars and walking people, and because the number and positions of the parked cars were constantly changing.

Our land mobile robot is equipped with ultrasonic proximity sensors, an infrared (IR) beacon reader, roll and pitch inclinometers, and the PRISM real-time stereo system (Nishihara 1984). Since general object recognition was not a central part of our research, we specified that the donor agent would be identified by a special IR frequency

emission. The overall strategy taken was to use the IR reader to determine the bearing to the donor, use the stereo system to determine the range, and then navigate to the donor using the ultrasonics to avoid obstacles. Because we wished to limit human involvement in the demonstration, the donor agent used was a Hero 2000 mobile robot rigged with an IR transmitter.

The Reaction Plan

Figure 1 shows the GAPPS reduction rules used to generate the reaction plan which accomplishes the find and fetch task described above. The goalem function at the bottom of the figure invokes the GAPPS compiler on the top-level goal. This goal is a conjunction of 1) a priority conjunction of goals and 2) a standing goal to always move towards the current navigation goal. A navigation goal is posted from a computed trajectory to a target. The priority conjunction of the form (prio-and $g_1, g_2, g_3, ..., g_n$) requires that all the goals be satisfied simultaneously, unless there are contradictory preconditions, in which case the plan should attempt to achieve the first n-1 goals, etc. Using the preconditions in the reduction rules and the conjunctions in the top-level goal, the compiler generates a total of 63 ways to accomplish the top-level goal.

By examining the rules, one can see that the competent behaviors being sequenced in the reactive layer are runaway, navigation, IR search, and PRISM (stereo) search. On each cycle of the resulting circuit[1], all the preconditions are examined to determine which set of behaviors will be active on that cycle. If "all goes as planned" this plan will first invoke the IR search activity until a bearing to the donor is found, then invoke the stereo search activity along the IR bearing until a range is found, and then, using the coordinates computed from the direction and range, invoke a navigation activity to reach the donor. Once the object has been received from the donor, the plan will invoke a navigation activity to return to the starting position. Being responsive to changes in the environment, however, this plan will, for example, also make sure the robot does not start the IR search if there are obstacles in close proximity, and will restart both the stereo search and IR search if the homing to the beacon is lost, e.g., if the Hero robot moves.

One should note that the rules for (maint find-donor-direction) and (ach find-donor-location) do not explicitly call for setting the direction or computing the location of the donor. These are handled in the (we-know-the-donor-direction) and (set-the-goal-as-the-donors-location) functions. If on previous cycles these functions return FALSE or a null goal respectively, on the cycles in which the perceptual searches are completed, the functions will return TRUE and will set the new navigation goal position respectively.

[1]Though the circuit cycle time is 0.1 seconds, the system cycles from sensors to actuators about three times a second.

The Behaviors

As stated above, the reaction plan allows the robot to accomplish its goal by interleaving the execution of the runaway, navigation, IR search and stereo ranging behaviors. No set of behaviors is committed to longer than one pulse of the circuit. In this way, the architecture achieves and maintains discrete states by managing continuous activity. The behaviors are discussed in more detail below.

The Runaway Behavior

The first behavior is an emergency behavior used for unexpected situations, such as the sudden appearance of an obstacle previously unseen due to specular reflection of the sonar pulses. This behavior is simply an energy minimizing routine (e.g., see Brooks 1986, Arkin 1987). The behavior takes the readings from a 24 sonar ring and a six sonar ring around the robot's waist and skirt respectively and thresholds them to detect readings which lie within the robot's safety radius (typically about 0.2 meters). Given danger readings, the behavior then generates a runaway vector which will drive the robot either forward or backward[2] away from the danger using an inverse square law summation of the sonar ranges. As is shown in the reaction plan (Figure 1), this behavior is called whenever there are sonar readings indicating that there is an obstacle in the robot's safety zone.

The NaTigation Behavior

The NaT (Navigation Template) navigation (or NaTigation) behavior is based on the use of Navigation Templates (Slack 1990a) for transforming a symbolic representation of the robot's surroundings into a drive/steer control vector. On each pass through the circuit, the behavior accepts the 24 sonar readings and the current goal position (relative to the robot's position) as input and uses that information to maintain a navigation plan for moving the robot to the goal position. The navigation plan is updated on each cycle, and thus the behavior will adapt to new goals, recasting current obstacle information into the new plan. The NaTigation behavior uses the sonar readings to maintain a model of the robot's current surroundings. Because the behavior is implemented in a precomputed circuit language (Rex), a compute-time limit is placed on the number of obstacles for which the system allocates memory resources. For the runs reported here, the system was compiled to maintain up to ten obstacles, which provided our robot sufficient awareness of the environment to accomplish all of the experiments discussed herein.

The obstacles are modeled as circles and each can contain up to five sonar readings, one from each of the last

[2]The robot employs a synchro-drive drive system; thus depending on the situation, the robot can either move forward away from the danger or back away from the danger which ever is the quickest way out of the situation.

```
(defgoalr (maint runaway)
  (if (anything-is-too-close)
      (do action (move-away-from-danger))
      (do anything)))

(defgoalr (ach move-to-goal)
  (if (or  (anything-is-too-close)
           (not (there-is-a-goal))
           (we-are-at-the-goal))
      (do anything)
      (do action (move-according-to-NaT-plan))))

(defgoalr (ach retrieve-widget-from-donor)
  (if (anything-is-too-close)
      (do goal (remember-the-current-goal-if-any))
      (if  (donor-has-been-located)
           (ach fetch-widget&go-home)
           (ach find-donor))))

(defgoalr (ach fetch-widget&go-home)
  (if (we-already-have-the-widget)
      (do goal the-robots-home-position)
      (do goal (set-goal-as-the-donors-location))))

(defgoalr (ach find-donor)
  (if (we-know-the-donor-direction)
      (ach find-donor-location)
      (maint find-donor-direction)))

(defgoalr (maint find-donor-direction)
  (if (ir-search-has-been-started)
      (do action (continue-search))
      (if (and (the-ir-search-is-finished)
               (the-returned-angle-is-no-good))
          (do rpt-ir-failure? true)
          (do anything))))

(defgoalr (ach find-donor-location)
  (if (stereo-ranging-is-started)
      (if (a-range-value-has-been-returned)
          (if (the-range-value-is-no-good)
              (do rpt-prism-failure? true)
              (do anything))
          (do action (continue-search)))
      (do action (start-stereo-ranging))))

(goalem (and (prio-and (maint runaway)
                       (ach retrieve-widget-from-donor))
             (ach move-to-goal)))
```

Figure 1. GAPPS psudo code listing of the reduction rules implementing the control structure. The do command sets a value in a vector of outputs. Goalem invokes the GAPPS compiler. Functions such as (move-away-from-danger) or (move-according-to-NaT-plan) generate appropriate values to be communicated to the robot actuators in the runtime environment.

five cycles of the circuit. Again, five readings was sufficient to allow the robot to accommodate the complexity of the environment in our experiments. On the sixth cycle the oldest reading is forgotten and a new reading may be placed into the obstacle representation. Those readings which when added would cause an obstacle to grow too large or which are at a great distance from the robot are not aggregated. This eventually leads to freeing up an obstacle resource (i.e., five cycles without a reading to add to the obstacle) which can then be used to model a new obstacle. Aggregation of new and decay of old information keeps the robot's model of the local surroundings up to date (Slack 1990b, Miller and Slack 1991).

A grouping algorithm arranges the circles into groups which represent single logical obstacles with respect to the width of the robot, i.e., the robot cannot pass between the circles of a group. Then each group is assigned a spin, i.e., a decision to pass by the group in a clockwise or counterclockwise manner, based on the geometric relationships between the robot, the goal and the group (see Slack 1992). The result is a qualitative plan for moving among the obstacles. For example, consider the situation depicted in Figure 2 taken from an actual outdoor run with our robot[3]. The goal position (donor robot) is in the upper left, and a car, moving at a greater relative speed, drove past the robot as indicated by the downward sloping vector. In the figure, the local obstacles fall into three groups: two trivial groups which the robot is to pass in a clockwise fashion and one large group resulting from the side of the car. The latter was initially assigned a clockwise spin because the first set of circles representing the car were detected to the right of the robot's path. However, as the car moved past the robot, the obstacle group representing the car starts to form a blockade, changing the relationship between the group of obstacles, the robot and the goal. At that point, the NaTigation plan guided the robot counter-clockwise behind the car. It is this ability to make real-

[3] In Figures 2 and 4, the positions of the robot and the objects are plotted from the telemetry of the robot's position and sensors.

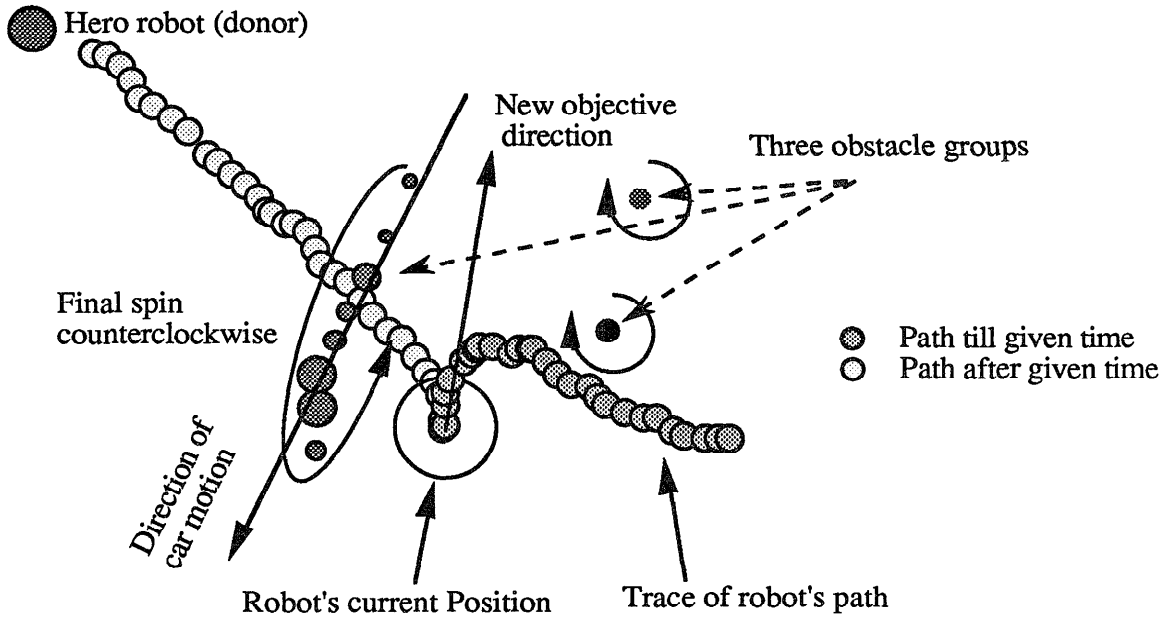

Figure 2. An example of the robot's reaction to a car moving across its trajectory

time navigation planning decisions which gives the NaTigation behavior its robustness.

IR Search

The basic IR search behavior is to turn the steer motor of the robot and look for IR intensity values values that indicate the presence of the beacon attached to the donor agent. The robot then stops the turn and makes a fine azimuth adjustment to compensate for overshoot. Our IR reader returns a status indicating the signal-to-noise ratio, a beacon number, a deviation angle in azimuth from the robot's current heading, a deviation angle in elevation (not used) and an intensity (not used).

The first phase of the search is to start a 2π counterclockwise turn, tracking the status code. When 1) a status indicating the best low-noise signal is read and 2) the azimuth deviation changes sign, indicating that the beacon emission has passed the center of the reader, the turn is stopped. If the final position of the robot is not within a set cutoff value, a final turn correction is executed which is the negative of the azimuth deviation. Then the bearing to the beacon is computed and returned.

The complexities in developing perception behaviors for our work center around robustness and control switching with the reactive motion of the robot. Since the runaway behavior can usurp the steer control from the IR behavior, the latter must be able to be interrupted and then resume the search. For example, to avoid starting a full turn after an interruption once the beacon has been found, the behavior remembers the robot orientation and position after each successful search. The behavior then uses that information to make either a clockwise or counterclockwise turn to make the shortest turn in the direction the beacon was last detected.

The routine is expected to be polled for the search continuously by the reaction plan even after the bearing is established. Thus, once the search is complete, as long as the IR emissions stay within the field of view of the reader, subsequent requests will yield the same value, or cause the routine to make small turn adjustments, thus providing a rudimentary tracking capability.

Stereo Search

The stereo vision system consists of two synchronized Vidicom cameras mounted on a Zebra pan-tilt-verge control head which feed video data to the PRISM-3 stereo/motion processor. PRISM-3 convolves the video frame with a Laplacian of Gaussian (LoG) operator to extract salient reflective features of the scene despite varying luminance conditions, reads six (software-specifiable) windows from the resulting video image of each camera, and cross-correlates each window of one camera with each of the other. A peak in the resulting array of 36 correlation measures indicates that the corresponding windows may be sampling from the same point in the scene. A disparity measure is computed to sub-pixel accuracy and is used to estimate the range to the point[4].

[4] Disparity is how far the cameras must be moved in x and y for the center of each camera to be focused on the same point in the scene. The range is the tangent of half the vergence angle times half the distance between the cameras.

The behavior starts the search by verging at the ground a short distance (usually six feet) from the robot's base in the direction of the steer motors. An expected range to the ground is computed from the height of the cameras and the angle of the downward tilt under the assumption that the ground is relatively flat. Errors in range are accounted for by the multiplicative and additive errors in camera height, separation, tilt angle, verge angle, and pixel size. From this point on, when the vision system moves to a new point, it will set its vergence based on the expected range to the ground. Any error which falls outside the range of accountable errors is determined to be a discontinuity from the flat ground plane, i.e., an object is detected (see Figure 3).

The cameras are panned left and right in increments that guarantee full coverage of the error envelope of the IR bearing result (usually +/- 2 degrees). The cameras are then tilted up in a step size determined by the minimum size of the object being searched for, and the panning is repeated. The process continues until a range discontinuity is found or a distance cutoff is attained. The ground distance to the discontinuity in the former case or an error flag in the latter case is returned as the result of the search.

The routine can be interrupted and restarted, but since that usually indicates that the bearing to the donor agent has changed, the behavior currently saves none of the information it last computed, and starts a new search on each new request. The behavior is competent because it can account for its own mechanical uncertainties, and because it can overcome the tendency of the head to stick in a down-looking position at steep tilt angles[5].

Performance and Limitations

The system was tested in both indoor and outdoor environments, but the following discussion focuses on the more difficult and thus more interesting outdoor environment. Outdoor environments are less structured than the indoors, and the number and types of light sources are not easily known or controllable. The area used for the experiments was a sloping parking lot behind one of the buildings of our organization, populated with cars, people and mobile trashcans which we could position for additional obstacles.

On clear days with some overcast, the robot carried out the find and fetch task robustly and repeatedly. It located the Hero, navigated safely among moving people and cars traveling at normal parking lot speeds and positioned itself within the reach distance of the Hero's arm (+/- 1.5 feet). The Hero then executed a simple reaction plan to place objects on the robot's platform using sonar ranging, and an

[5] This sticking was due to debris in the gears which affected small angular motion. The behavior tracks expected encoder readings from the motors and invokes a series of increasingly involved un-stick routines, the last being "ask a human for a helpful tap".

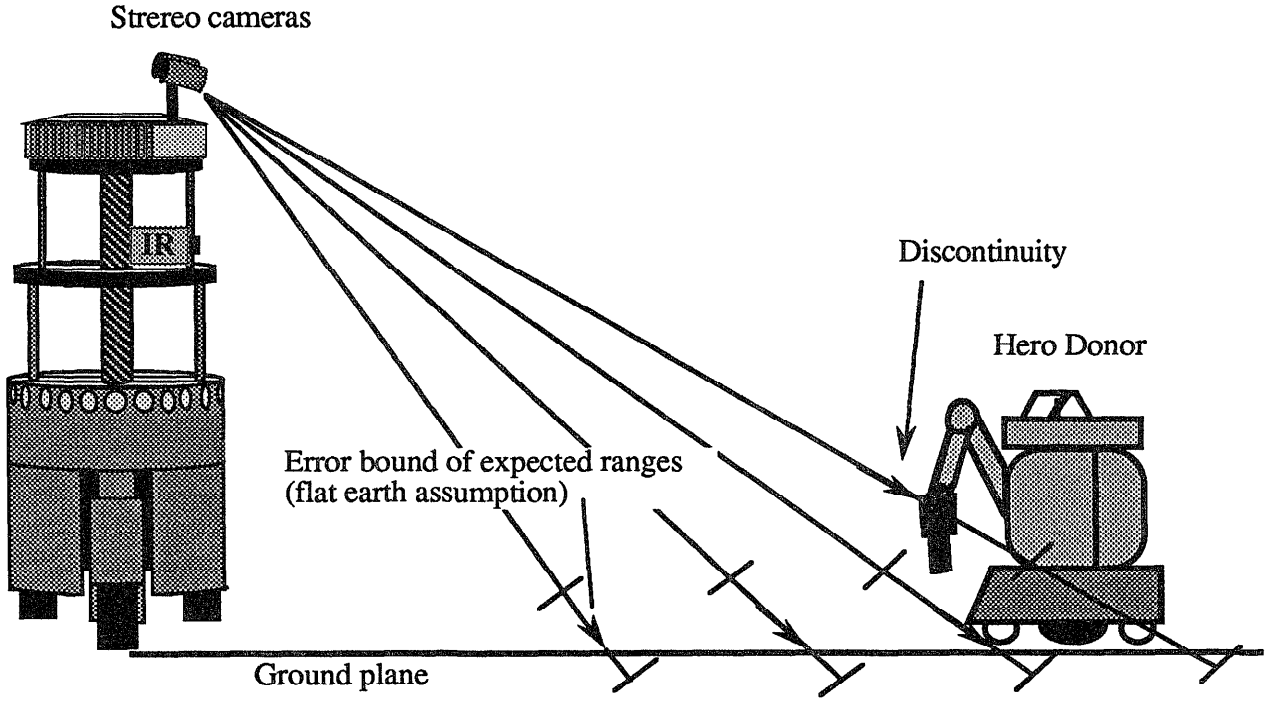

Figure 3. Using stereo vergence at expected ranges to detect objects on the ground plane

object-on-board keyboard signal was transmitted to the robot to indicate the object was on board. Then, as shown in the reaction plan of Figure 1, the robot navigated back to its starting position (the remembered starting position becomes the new goal position after the widget is on the platform) where it would station-keep awaiting the next command. As the reaction plan indicates, toggling the object-on-board command caused the robot to alternately navigate to the donor and to the starting position.

Figure 4 depicts a typical run. For a donor position 25 feet from the retriever, the total time required to navigate the round trip course was between three and five minutes, depending on the number of obstacles to be negotiated. With IR and stereo search the total task completion time was between ten and twelve minutes, depending on the number of times dynamic obstacles interrupted the sensor searches.

NaTigation and Runaway

The first two goal reduction rules of Figure 1 above, show that the NaTigation and runaway behaviors are in an exclusive-or relationship, with the runaway behavior taking precedence in the case of a conflict. Due to the robustness of the navigation approach, during execution the runaway behavior is rarely activated, i.e., only when an object suddenly appears in danger proximity to the robot in one cycle. For example, during the run depicted in Figure 4, the runaway behavior was activated only once due to the fact that we deliberately rolled a trashcan within a few inches of the robot in order to block its path. This resulted in the robot backing away from the obstacle until the obstacle was beyond the danger radius and the runaway behavior thus became inactive. At that point, the navigation plan was modified to account for the obstacle and the robot proceeded smoothly towards the goal avoiding the obstacle[6] .

The limitation of the current approach results from the fact that while the navigation and runaway behaviors are robust within their local scope there is no global memory. As a result, the robot can become trapped in a box of obstacles which number more than the allocated memory resources. This rarely happened in our test runs, but see the conclusions section for our ideas on solving this problem.

---

[6] For the example reported here, the robot's maximum speed was 0.2 meters per second. The speed is set to be a function of the robot's distance to the nearest obstacle as well as how well the robot was oriented with respect to the guidance from the NaTigation plan. Given these settings the robot was able to traverse to the Hero in well under two minutes.

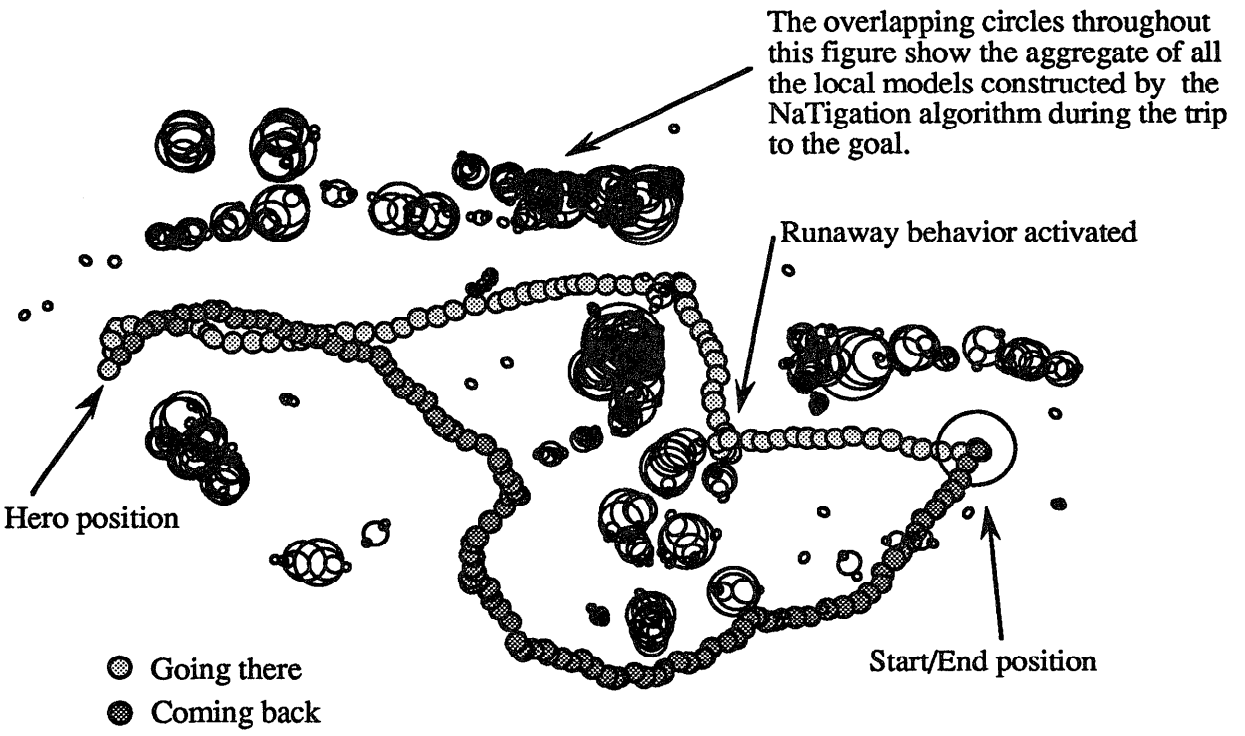

Figure 4. A typical parking lot run

Sensor Searches

The outdoor limitations of the IR search routine were restricted to the sensor itself. The maximum range we were able to achieve was 25 feet. When the sun was in the background of the transmitter, even on cloudy days, the signal to noise ratio was too low to get reliable readings; a person was needed to serve as a sun block by standing behind the Hero in those instances. When wind gusts rocked the Hero, making the support to the transmitter unsteady, or when too few readings were obtained to sense the plus-to-minus change in azimuth angle, the reaction plan would appropriately restart (retry) the search .

The stereo search had to deal with variable luminance conditions due to shadows, cloud cover, and time of day. As well, the flat earth assumption was violated with respect to the zero-slope condition we previously used indoors, and the ground surface (macadam) is coarser than our indoor carpet surface. We manually reset the camera's f-stops for the luminance conditions, but to deal with the coarser ground texture, we added an automatic LoG size control which decreased the size of the convolution operator with respect to increased ranging distance. Subsequently, the surface coarseness served to provide more textured features, thus improving performance by reducing false correlations. The ground slope limited the effective range of the search only when the robot itself was situated in an area with slope different than that in the visual search path.

Related Work

This work can be compared most directly with recent efforts on the JPL Rover (Miller and Slack 1991). That effort used Reaction Action Packages (RAPs) (Firby 1989) for high-level goal representations and low-level behaviors written in the Alpha circuit language (GAT 1991). Extensive runs of a detailed simulation were conducted along with a partial outdoor run with the actual system, all of which showed the Rover able to negotiate rugged terrain and move robustly to a series of waypoints. A stereo system (Matthies 91) was the primary sensor used to feed a NaTigation planner. Our work shows comparable results using reaction plans and layered competences.

The navigation approach is somewhat similar to the gradient summation work of Arkin and others (Arkin 1987), wherein obstacles act as repulsors and goals act as attractors. A major drawback to gradient summation approaches, however, is that there are no plan semantics for making motion decisions (i.e., whether to pass to the right or to the left of an object) based on the strategic goal. Thus, the algorithms often leave the robot oscillating in a local minima. Our approach essentially uses knowledge of the strategic goal to prevent the local minima situations from developing.

PRISM-3 is one of a line of stereo-vision processors designed and built by Nishihara (Nishihara 1990) and is an outgrowth of the work on stereo undertaken by David Marr at MIT in the late 70's and early 80's (Marr 1982, Nishihara 1984). There is a wealth of related work on active vision (see (Ballard 1991) for an overview), some of the most relevant being the building of visual routines using stereo (Coombs and Brown 1991), and the use of visual routines for auto-calibration of the vision system (Bajcsy 1988). Our work is the first we are aware of wherein task-specific routines were developed using frame-rate stereo ranging as a competence.

Conclusions and Future Work

Our experiments have shown that the architecture can scale to more complex tasks given the appropriate granularity of operators, and that the resulting robot behavior smoothly accommodates natural changes in the environment. Because of this, previously reported benefits of this approach -- use of robust behaviors, high-level goal representation, consistent semantics, and ease of programming -- take on a stronger significance. We conclude with a discussion of a number of directions to pursue in this work. They include extending the existing behaviors and increasing the capability of the overall find and fetch reaction plan.

Because the current stereo search behavior searches out from the robot base along the ground plane, there can be no obstacles between the robot and the donor until this search is completed. This limitation can be changed by searching down from the horizon toward the robot base. Also, though we can improve upon the 23-25 foot range limit of the vergence control, there are fundamental limits to this ranging approach, and longer range searches should be based more directly on shape analysis from the image plane. Ultimately, we want the vision system to recognize as well as determine the distance to the donor agent.

The IR range limitations suggest extending the IR search routine to navigate about the area in question looking for IR emissions. This will also allow the robot to find multiple donors to retrieve multiple objects. A solution to the box-canyon problem with the navigation mentioned earlier is to remember configurations of obstacles, thus allowing the construction of a global topological model. Using this information a planning system could adjust the robot's current goal in order to lead the robot out of trouble (see e.g., Krogh and Thorpe 1986).

A more fundamental direction of research which we are pursuing involves exploiting the high-level goal representation to allow for a deliberative planning capability. Our current version of GAPPS provides for STRIPS-like operators to be used by the compiler to regress reaction plans (Kaelbling 1990). These operators provide a representation than can serve as a basis for integration with an asynchronous planning system. Having the deliberative capability would allow the robot to generate new reaction plans to cover situations for which it has no response or when it cannot make useful progress in a reasonable amount of time.

References

Arkin, R. C. 1987. Motor Schema Based Navigation for a Mobile Robot: An Approach to Programming by Behavior, in the Proceedings of the IEEE International Conference on Robotics and Automation, Raleigh, North Carolina, pp. 264-271.

Bajcsy, R. 1988. Active Perception, Proceedings of the IEEE, Vol 78, No. 8, August.

Ballard, D. H. 1991. Animate Vision, *Artificial Intelligence*, v.48, p. 57-86, Elsevier Press,.

Bonasso, R. Peter. 1991a. Integrating Reaction Plans and Layered Competences Through Synchronous Control, In Proceedings of the 12th International Joint Conference on Artificial Intelligence. Sydney, Australia, August. Morgan Kaufman.

Bonasso, R. Peter. 1991b. Underwater Experiments With A Reactive System For Autonomous Vehicles, In Proceedings of the 9th National Conference on Artificial Intelligence. Anaheim, CA., July . AAAI Press.

Brooks, Rodney A. 1986. A Robust Layered Control System for a Mobile Robot, *IEEE Journal of Robotics and Automation*, RA-2:14-23, April.

Coombs, D. J. and Brown, C. M. 1991. Cooperative Gaze Holding in Binocular Vision, *IEEE Control Systems Magazine*, June.

Firby, James R. 1989. Adaptive Execution in Complex Dynamic Worlds. PHD Diss., YALEU/CSD/RR #672, Yale University.

Gat, Erann. 1991. Taking the Second Left: Reliable Goal-Directed Reactive Control for Real-World Autonomous Robots, Phd Diss., Dept of Computer Science, VPI.

Kaelbling, Leslie Pack. 1988. Goals As Parallel Program Specifications. In Proceedings of the Seventh National Conference on Artificial Intelligence, pages 60-65, Minneapolis-St. Paul, Minnesota, 1988. AAAI Press.

Kaelbling, Leslie Pack. 1990. Compiling Operator Descriptions into Reactive Strategies Using Goal Regression. TR90-10. Teleos Research. Palo Alto, CA.

Krogh , B. H. and Thorpe, C. E. 1986. Integrated Path Planning and Dynamic Steering Control for Autonomous Vehicles, in the Proceedings of the International Conference on Robotics and Automation, San Francisco California, pp. 1664-1669, April .

Marr, D. 1982. *Vision*, W. H. Freeman and Co., NY.

Matthies, H. L. 1991. Stereo vision for planetary rovers: Stochastic modeling to near real-time implementation, Technical Report D-8131, Jet Propulsion Laboratory, Pasadena, CA, January.

Miller, D. P. and Slack, M. G. 1991. Global Symbolic Maps From Local Navigation, In Proceedings of the 9th National Conference on Artificial Intelligence, Anaheim, California, July. AAAI Press

Nishihara, H. K. 1984. Practical real-time imaging stereo matcher, *Optical Engineering*, 23(5):536-545, October.

Nishihara, H. K. 1990. RTVS-3: Real-time binocular stereo and optical flow measurement system. System description manuscript, Teleos Research, Inc., July.

Rosenschein, Stanley J. and Kaelbling, Leslie Pack. 1986. The Synthesis of Digital Machines with Provable Epistemic Properties. In Proceedings of the Conference on Theoretical Aspects of Reasoning About Knowledge, pages 83-98. Morgan Kaufman.

Schoppers, Marcel J. 1989. In Defense of Reaction Plans As Caches. *AI Magazine*, 10(4): 51-60.

Slack, Marc G. 1990a. Situationally Driven Local Navigation for Mobile Robots, Ph.D. Diss., Dept. of Computer Science, VPI, May, also see JPL Pub. 90-17, NASA.

Slack, Marc G. 1990b. Coordinating Sensing and Local Navigation, In Proceedings of SPIE's 9th Conference on Sensor Fusion, pages 459-470, Boston, November.

Slack, Marc G. 1992. Computation Limited Sonar-based Local Navigation, In Proceedings of the AAAI 92 Spring Symposium on Selective Perception, March. AAAI Press.