REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE DATE SUBMITTED: November 8, 1995	3. REPORT TYPE AND DATES COVERED FINAL TECHNICAL (DATES COVERED) 4/1/91 - 12/31/94

4. TITLE AND SUBTITLE

PROJECT TITLE: TOWARDS INTELLIGENT AUTOMATED FORCES FOR SIMNET

5. FUNDING NUMBERS

ONR GRANT NO. :
N00014-[redacted]-1624
91-J
ACCOUNT NO. :
53-4540-8586

6. AUTHOR(S)

P.I./P.I.'S INVOLVED: PAUL ROSEBLOOM

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

UNIVERSITY OF SOUTHERN CALIFORNIA/
INFORMATION SCIENCES INSTITUTE
4676 Admiralty Way
Marina del Rey, California 90292
Suite 1001

8. PERFORMING ORGANIZATION REPORT NUMBER

USC REPORT NO.
(IF ANY)

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

SPONSOR PROGRAM MONITOR: OFFICE OF NAVAL RESEARCH

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE: DISTRIBUTION UNLIMITED

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

19951121 045

Accesion For	
NTIS CRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and / or Special
A-1	

14. SUBJECT TERMS	15. NUMBER OF PAGES 10
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Final Report

Towards Intelligent Automated Forces for Simnet
ONR Grant No. N00014-91-J-1624
For the Period 4/1/91 - 12/31/94

Paul S. Rosenbloom
University of Southern California
November 8, 1995

This is the final report on an effort to perform seed research towards the ultimate creation of automated intelligent forces (IFORs) for SIMNET. SIMNET was a large-scale combined-arms battlefield simulator developed jointly by Bolt Beranek and Newman, (BBN) Inc. and Perceptronics, Inc. under the sponsorship of the Defense Advanced Research Projects Agency (DARPA) and the United States Army (Crooks *et al*, 1990, Jacobs, McDonough, & Crooks, 1990). The simulator provided an opportunity for training fully manned units – at the platoon, company, and battalion levels – through direct engagements against comparable opposing forces. In addition to serving an important training function, the intent was also to use SIMNET as a testbed for developing and assessing doctrine and tactics, and as an aid in evaluating proposed weapon systems prior to their construction and acquisition. All of this was feasible because SIMNET was constructed from a set of inexpensive "selective fidelity" physical simulators that were dispersed across the country, and were networked together to provide joint engagements.

The addition of semi-automated forces (SAFORs) to SIMNET provided a means of increasing the number of units involved in an engagement without the expense of additional physical simulators and the troops required to man them. With SAFOR, a single human commander was put in direct charge of an entire simulated unit – a platoon, company, or battalion – in such a way that no other humans were needed at any of the lower command levels. The human commander could selectively move his focus up and down the command chain, immersing himself in the information environment appropriate to the selected role, and directly controlling the corresponding unit(s). The automated portion of SAFOR consisted of *behavioral scripts* that delineated how units should behave in well-circumscribed situations – the key decisions at all levels were still made by the human commander, through the issuance of fragmentary orders to the SAFOR system.

More recently the SIMNET environment has been replaced by the Distributed Interactive Simulation (DIS) Environment, and the BBN SAFOR by the Loral ModSAF (Calder *et al*, 1993). However, the basics remain the same.

Our overall goals in this area are (1) to perform research and development towards generic technology for the creation of human-like intelligent agents for synthetic environments, specifically to include intelligent forces for synthetic battlefields; and (2) to create a range of specific intelligent forces (IFORs) for deployment in synthetic battlefields. These intelligent agents must be able to use knowledge about their environment, about other agents, and about missions, doctrine and tactics, to behave in a rich three dimensional world, to plan when necessary, to communicate and coordinate with other forces, and to learn about their environment and their behavior in it. Moreover, all of this must be integrated together so as to produce real-time human-like behavior. The intent is to provide inexpensive but realistic force supplements for large-scale synthetic exercises.

The core of the technology underlying the development of these IFORs is the Soar architecture, which supports forms of integrated reaction, planning, and goal-based problem solving; expert-level performance; learning from experience; interaction with external environments; natural language processing; agent modeling; and psychologically realistic behavior (Rosenbloom *et al*, 1991, Rosenbloom, Laird, & Newell, 1993).

The actual effort under this grant focused on seed research towards the construction of such Soar-based IFORs.[1] This covered only a small fraction of the total effort required to build real IFORs, but it was still sufficient to investigate several of the key issues, and to provide additional hard evidence as to Soar's plausibility as the basis for real IFORs.[2] This work was pursued in the context of a simplified SIMNET-like environment – called *GroundWorld* (originally *GridWorld*) – developed at the Hughes AI Center. GroundWorld is a two-dimensional real-time world in which both space and time are represented as continuous quantities (but computed discretely). The domain can include multiple teams of opposing and cooperating agents – each with limited perceptual abilities and the ability to move around and control a gun barrel (including the ability to shoot at other agents) – walls behind which agents can hide, and terrain of varying degrees of roughness.

Our initial plans were well summarized by the following statement of work, extracted from the original proposal:

- **First Year:**
  1. Port the GroundWorld to the Sun (it currently runs on an Apple Macintosh).
  2. Interface Soar to the GroundWorld as the controller of a single agent.
  3. Demonstrate single agent planning and execution in the environment (in the presence of simple non-Soar-based opposing agents).
  4. Demonstrate skill acquisition in the context of learning to become increasingly reactive to environmental contingencies.
- **Second Year:**
  1. Demonstrate the ability to cope with unexpected environmental circumstances by a combination of replanning and skill acquisition.
  2. Augment the agent to take a simple set of commands in restricted english.
  3. Add a second agent that is under the command of the original agent.

We ended up completing everything through item 1 of the second year, before refocusing the remainder of the effort on spatial reasoning. This refocusing occurred for two reasons. The first reason was that the large (ARPA-funded) project for which this was intended as seed research was now in full gear, and was able to put considerably more effort into construction of broad-

[1]With one small philosophical digression on the nature of symbolic architectures and the organization of intelligence (Rosenbloom & Newell, 1993).

[2]For a recent summary of the larger IFOR effort that grew out of this seed research see (Tambe, Johnson, Jones, Koss, Laird, Rosenbloom, & Schwamb, 1995).

based (and realistic) intelligent forces (for, at that point, Navy tactical air combat). Though much was learned from this (ONR-funded) effort for (and transferred to) that larger project, it thus made sense to refocus this smaller effort on a key issue that was not yet being addressed in the larger effort. The second reason was that symbolic spatial reasoning kept showing up in our early efforts here as the key missing ingredient in providing flexible and effective behavior in Groundworld.

The remainder of this report describes the progress made on this seed effort, starting with porting GroundWorld to the Sun and interfacing Soar to it; continuing with initial (primarily reactive) agent construction, followed by agents that combined forms of spatial and cognitive reasoning (and learning); and wrapping up with the beginnings of an in-depth investigation of spatial reasoning.

Port the GroundWorld to the Sun

As we received it, the GroundWorld simulator ran on an Apple Macintosh, so the first task was to convert it to the Sun and Unix, using X-Windows for the graphics (via Sun's Lispviews). New graphics capabilities were added, including a global picture of the battlefield (instead of just a tank's point of view). Bottlenecks were identified and removed resulting in a three-fold improvement in processing time. Other enhancements included an interactive capability for stepping through the simulation, and a number of improved algorithms for line-of-sight calculations and more realistic directional sensing.

Interface Soar to the GroundWorld

A simulated tank in GroundWorld receives perception and effects commands via a user-written control procedure which is called periodically in the simulation loop. Interfacing GroundWorld and Soar consisted in writing an I/O module which converted these perception inputs and command outputs to/from Soar's working memory. This involved some choices as to how much detail should be filtered out by the I/O module and in what form the information should be presented to Soar's working memory. The general principle followed was that arithmetic quantities and calculations reside in the I/O module and only symbolic information should reside in Soar's working memory.

Initial Agent Construction

Agent construction was approached by building a succession of more sophisticated GroundWorld tank controllers in Soar. The actual control of a tank was accomplished by writing Soar productions. Emphasis was first placed on programming low-level behaviors which enabled the tank to successfully move from a start position to a goal position, avoiding obstacles and other tanks along the way. Initially, the tank was purely reactive in that it had no internal representation of the world and hence no plan to follow; instead, its behaviors were essentially mappings from its perception and state to command outputs. The behaviors included turning and moving towards ends of visible walls, avoiding enemy tanks and following a fixed distance behind a friendly leader tank, shooting at visible enemy tanks, avoiding obstacles sensed at close range, and moving towards its goal. Because these behaviors can sometimes conflict, arbitration was needed to choose between them: this was accomplished by Soar's preference scheme and support for interruption and resumption of operators. A model of behaviors as continuous operators which persist over time was developed and led to new ways to regard Soar's decision cycle.

This simple approach enabled the tank to accomplish some goals in (simulated) real time, but the result was a very primitive form of "intelligence". One aspect of this was a lack of navigational capabilities – the tank had no representation of a global map, and hence no ability to use such a map to plan efficient routes. This led to two investigations: how to enable the agent to learn a map from its reactive exploration, and how to then use this map to navigate more efficiently in a dynamic world where the map could not be guaranteed to be accurate. The first aim was accomplished by having the agent deliberately remember locations where it perceived landmarks and took actions. These perception-action procedures were then massaged (by the agent) into a procedural map which was then followed in a more efficient manner because many of the subgoal decisions were eliminated and some of the paths traversed were shortened. Since this map abstracted out local obstacle avoidance motions, and those behaviors were still active, the agent was still able to navigate in a dynamic environment while following the map. A more expressive map representation was also investigated which allowed the agent to answer self-queries concerning routes in the terrain via an internal simulation. This internal simulation paved the way for more complete planning capabilities in reasoning about the problem of moving around a dynamic uncertain world.

Integrating Cognitive and Spatial Reasoning

One of the greatest strengths of an integrated AI architecture, such as Soar, is that it should be able to facilitate the construction of agents that combine a wide range of knowledge and reasoning methods. On the other hand, one of Soar's greatest weaknesses is its lack of demonstrable abilities to use particular types of knowledge and reasoning, such as spatial knowledge and reasoning, that are critical to effective agent performance in worlds such as Simnet and GroundWorld (or the real world, for that matter). Our goal for this segment of the effort was therefore to understand how spatial reasoning should be done in Soar for the GroundWorld. Our focus was not on inventing new more optimal algorithms for well studied problems such as shortest-path navigation. Instead, we focused on combining a range of spatial and path planning capabilities that varied according to available knowledge, tasks for which they were appropriate, and performance criteria; and on their interactions with the rest of the cognitive system. This is a topic that has been little studied so far, and also one in which Soar can potentially provide great leverage.

To drive this effort we started with a class of safe-traversal tasks which stressed a combination of spatial and non-spatial cognition. The tasks all involved two agents – one friendly agent (referred to as *R2*) and one hostile agent – in a two-dimensional world that included walls which blocked both movement and vision. R2's task was to travel between two points in the world without being caught. The hostile agent's task was to catch R2. Only R2 was Soar based; the hostile agent was a simple stateless agent that wandered around aimlessly until it saw R2, chased it until it caught it or could no longer see it, and then wandered aimlessly again if it could no longer see it.

An example trace of R2's behavior can be seen in Figure 1. In this figure, R2 initially followed a plan based on criteria of safety and length. Once it perceived the hostile agent – while at (approximately) location [450, 500] – it reacted immediately by starting to run away, while simultaneously selecting a hiding place (behind a wall) and generating a plan by which to get there. Once safely in its hiding place, it replanned a route to its goal – making use of rules learned during the original planning process – and once more proceeded on its way.

In general, R2 began by planning a route from its initial position to its goal position. It did this with the aid of a map in which the walls were used to guide the decomposition of the world

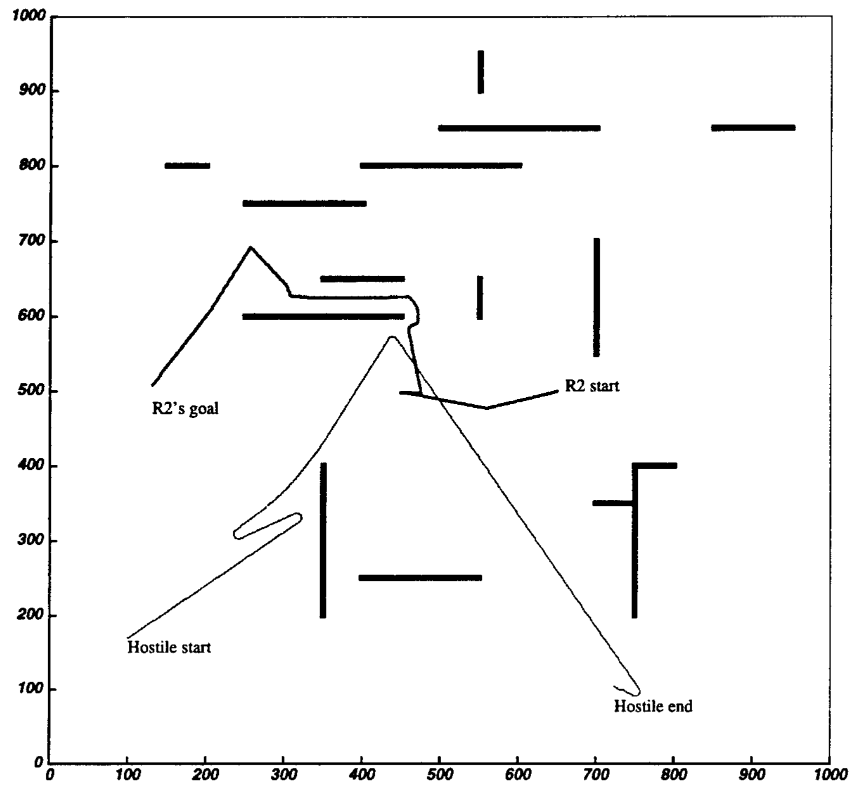

Figure 1: Behavioral trace of interactions between R2 and the hostile agent.

into rectangular regions of free space. The search for a path took these regions as primitive nodes, and heuristically looked for a short, safe path. Shortness was determined by the obvious path-length metric. Safety was determined by a set of heuristics about possible ambush points; that is, about ends of walls behind which R2 couldn't see. The search proceeded depth first, using a greedy heuristic based on the distance to the goal to order the choices at each point, and backtracking whenever either an unsafe node or a dead-end was reached. In the process, Soar's learning mechanism stored away new rules that captured the results of searching this tree of paths.

It then followed this path until the hostile agent was detected. Immediately, a reactive behavior was triggered off by a production sensitive to this detection. This behavior caused R2 to run away from the hostile agent. While running away, R2 used a set of heuristics based on the relative location of the hostile agent and the near-by walls to select a set of potential hiding places that it hoped to be able to reach before it was caught. It then planned paths to these hiding points, but using a very different strategy than during its initial path planning. Here it was important that the planning proceed quickly, and that the plan get the agent to its hiding place quickly. Issues of safety – at least with respect to possible ambush points – no longer mattered, while the ability to take short-cuts mattered a great deal. Furthermore, these short-cuts needed to be sensible; that is, they should not have reduced the distance between R2 and the hostile agent. Given these constraints, especially the need for a quick response, the hiding planner relied on pre-computed abstract plans. These abstract plans were instantiated for all of the potential hiding positions. Some of these plans were immediately rejected because their preconditions – which tested for locations of different walls – were violated. The remaining plans were then heuristically ranked, and the best one was selected.

Each of the abstract plans had failure conditions associated with them. Based on simple spatial reasoning, the failure conditions tested if a plan was doomed to fail. If so, that plan was abandoned, and the planner went back to its set of abstract plans to create and execute a new plan.

The abstract plans allowed R2 to move very close to walls, unlike the plans generated by the initial planner, where the path moved through centers of regions to avoid possible ambush points. Moving close to walls allowed R2 to take short cuts to the hiding spot. But these short-cuts extracted a price – the uncertainty in the movement of R2 could cause it to bump into nearby walls. The problem became particularly severe when the walls had small uncertainties in their lengths (as they did in some of our experiments.) To avoid this, R2 had a built-in reactive wall-avoidance procedure, based on a *potential field*. A field was set up in which the obstacles repelled R2, thus allowing R2 to avoid bumping into walls. It was this reactive behavior that caused the curved motion in R2's path around the wall. Note that in general, the potential-fields approach is problematical due to the presence of local minima. However, by using it in conjunction with the hiding planner, the problem of local minima was avoided.

Once the agent was safely hidden, it replanned a path to its goal. (In many cases a good deal of what was learned during the initial planning process could transfer to this replanning process.) If the agent could now get safely to its goal, it would. However, if it once again met up with the hostile agent, it would again try to hide and then replan once more.

When considered as a whole, R2 demonstrated a close integration of reaction, planning, execution, interruption, replanning, learning, search, use of heuristic knowledge, navigation, and abstract spatial reasoning (about hiding places). It used maps and (simulated) vision to provide spatial information, and varied the nature of the spatial reasoning and the path-planning criteria dynamically as a function of the situation. In many ways the behavior was still rather simple and special purpose – partially reflecting the fact that the opponent was itself quite simple – but it did provide a broad system that could form the basis for increased sophistication.

Some weaknesses in the design included R2's need for a reasonably accurate map (it could not survive if no map was available), its inability to learn the map, its lack of any concept of time, its inability to form and use high-level strategies (other than its one current strategy of hiding), and an incomplete integration of vision and planning.

Towards Flexible Spatial Reasoning

We began to investigate issues of flexible spatial reasoning with the goal of allowing a mobile autonomous agent to plan and follow routes according to several possible criteria and constraints (such as safety, speed, and distance) using diverse sources of knowledge that may or may not exist (maps, perceptual input, information about opponents, etc.). Our approach was a hybrid one because, though each existing technique had its strengths, none seemed up to the full range of criteria and knowledge. We combined low-level reactive behaviors (for avoiding obstacles, running away, and heading to a destination) to control short-timescale behavior, an algorithmic "black-box" route-planning algorithm to take advantage of the state of the art in this area, and a set of more flexible planning and replanning capabilities within the intelligent agent itself (Stobie, Tambe, & Rosenbloom, 1993). We later expanded the set of algorithmic "black-box" route planners available by coding some additional ones ourselves, and acquiring others from other researchers (in particular, potential fields, behavior arbitration, weighted grid/pixel searched by A*, visibility graph, and the centroid method). This resulted in a route-planning testbed which was very useful for examining the strengths and weaknesses of current approaches. (In the process we also ported the Groundworld environment to work with the new, and much faster, C-based version of the agent software.)

The question of how to integrate together such a disparate set of route-planning capabilities has not yet been addressed in robotics because current robots tend to perform simple tasks in uniform environments. Initially we addressed this integration question via a lookup-table of planners indexed by pre-identified conjunctions of criteria. However, the shortcomings of this approach then led us to construct a prototype which chose planners based on an explicit representation of their characteristics. Although this method was an improvement, in that the agent had some declarative knowledge about the capabilities of the available route planners, we began to realize that the crucial problem was the gap between task-level requirements and geometric-level concepts. To take just one example, safety from possible ambush is a task-level requirement that was used implicitly by one of our symbolic, heuristic route planners. Determining safety from possible ambush requires an understanding of what it means in geometric terms, such as obstacle shape, size, and position, and lines of sight; for example, there is little room to escape when moving towards a wall when an enemy may be behind a side wall and the other side is blocked by another wall. However, this reasoning that relates safety to geometric concepts was compiled into the planner's heuristics, with the result that the agent itself had no knowledge of the connection between safety, mobility, and geometry. This gap meant that the agent could not extend its competence to different shaped obstacles or regions, to terrain with hills or marshes, or even to different sensorimotor capabilities for acceleration and vision.

In order to study how to bridge this gap, we then asked what knowledge must an agent have to flexibly plan routes for new situations and acquire new strategies for movement. Existing systems that learn to improve their route planning as a function of their environment tend to use low-level non-symbolic representations of their environment and route, and to use experimental techniques such as reinforcement learning that tune parameters as a function of successes and failures in the environment. In contrast, we were looking at higher-level symbolic representations and analytical (rather than experimental) learning techniques. The symbolic representations were critical for the agent to be able to use the knowledge flexibly; for example, not only should it have been able to use its knowledge to plan a route on its own, but it should also have been able to understand how to use an existing tool (such as a specialized A*-based planner) to help generate a plan (including how to adapt whatever is produced by the tool), and to explain why the route is appropriate (with respect to the agent's goals and the environment). The use of symbolic representations, plus the ability to explain behavior, provided the opportunity to utilize analytical learning techniques (such as explanation-based learning) that could acquire generalized knowledge from single examples, rather than necessarily requiring many examples/experiments to support induction of generalized knowledge.

Our strategy was to identify a tractable subset of synthetic planar geometry as the underlying spatial theory, and to build a prototype system – a Meta Route Planner (MRP) – which used this theory to generate and learn route planning operators, and to use existing route planning algorithms. Our methodology was to examine incrementally more difficult scenarios, identifying what geometric knowledge was required to achieve them by asking what explanations would be acceptable. As an example of an acceptable explanation, we wanted our agent to justify why it was moving close to an obstacle by relating the task criteria (e.g., find the shortest route) to geometric knowledge (e.g., shortest paths pass through vertices and edges of polygonal obstacles).

The first result of the investigation was a language (implemented in Soar) which provided a very expressive means for specifying route planning problems. It was based on an ontology of concepts involved in navigation, consisting of general mathematical objects such as sets and functions, geometric objects such as points and lines and domain objects such as agents, maps, route problems and optimal traversal criteria. Given a route planning problem, the challenge facing MRP was to match the problem to a suitable (black-box) program. This corresponded to representing a route planning program as a triple of structures, namely a (variabilized) path problem, an argument list and a program. When the program's problem structure exactly matched an input problem, the argument list was instantiated, the program was run and the agent had its required route. The major difficulty was when no method's problem structure exactly matched the input. Cast in these terms, the research emphasis then became to develop the problem solving methods and domain-specific knowledge which would generate (a combination of) route planning program instantiations that matched an input problem.

The overall problem-solving method was means-ends analysis to reduce the difference between problem substructure and available route planners. Several sources of knowledge were identified and partially implemented (in Soar) to reduce the differences, compose programs and adapt their results. The ontology itself yielded a number of (weak) methods from the is-a and part-of structure of objects, such as abstraction and divide and conquer, respectively. For example, a path could be considered as simply a set of locations (if order of visits is immaterial to the problem) or as a sequence (if continuous information is immaterial) or in general as a function; since a set is a kind of sequence which is a kind of function, there is a natural organization and control of knowledge. Some spatial predicates themselves suggested a search strategy, such as "through" suggesting composition of subproblems and "avoid" suggesting generation of solutions and testing non-intersection. Since optimality criteria for A* grid-based programs could be represented as aggregates of functions on locations and directions, this led to the agent being able to design its own evaluation functions in a truly autonomous manner. Finally, since an agent's sensorimotor capabilities were represented in the ontology and included in problem descriptions, MRP could successfully design simple behaviors for executing routes.

The general structure of the problem solving method in MRP was not new, bearing similarities to conjunctive planning, analogy (specially case-based reasoning) and automatic software design. Instead, the contribution of this research was the tight integration between a content theory of a domain and the problem-solving agent. In contrast to other mixed-method systems, the different reasoning strategies were not separate modules to be invoked by an overall controller, but rather arose naturally from the way domain knowledge was represented in Soar problem spaces and operators. For example, a street map consisted of a number of routes, or solutions to planning problems, and thus could be used in the same way as route planners. In contrast to case-based approaches, the problem solving and representation was uniform across maps and planners.

Finally, evaluating this research is still an open question: how complete and useful is the set of navigation problems covered by MRP? In seeking an empirical answer, the intention (outside of this Grant) is to capture the required knowledge for three different task domains. The first is the pursuit-evasion task in the Groundworld simulation, where the objective will be to emulate the hand-crafted knowledge developed in this research project in the Groundworld-Soar system. The environment is continuous and criteria include escape from a moving pursuer. The second domain is city route planning, which involves a mixture of discrete and continuous spaces and knowledge about trading off longer distance against higher speed on freeways. The third domain is open terrain planning by a low flying helicopter – a key aspect of our current ARPA-funded IFOR development – which requires reasoning about visibility and elevation. It is expected that this diverse mixture of tasks should reveal the strengths and weaknesses of the MRP approach.

Conclusion

The goal of this effort was to perform seed research towards a larger effort on Soar-based intelligent forces for SIMNET. In this seed effort, we ended up being able to construct simple agents that combined reactivity, planning, execution, replanning, navigation, learning, use of knowledge, and abstract spatial reasoning. The agents were also able to use simple maps, and to select among planning methods as a function of the situation.

Beyond this simple agent construction we also developed the beginnings of a theory and implementation of the role of general spatial knowledge in navigational behavior. Though more work is clearly called for here, it provides a interesting beginning.

Beyond the scope of this Grant, we were able to leverage some of what we learned to the development – under funding from ARPA, NRL, and NRaD – of complex intelligent forces for both fixed wing and rotary wing aircraft in the DIS environment. These forces have already participated in an operational military exercise (Simulated Theater of War – Europe; AKA STOW-E) as well as in one other significant synthetic engagement (Engineering Demonstration 1; AKA ED-1) on the path to the STOW-97 ACTD.

References

Calder, R. B., Smith, J. E., Courtemanche, A. J., Mar, J. M. F., & Ceranowicz, A. Z. (1993). ModSAF behavior simulation and control. *Proceedings of the Third Conference on Computer Generated Forces and Behavioral Representation*. Orlando, FL: Institute for Simulation & Training.

Crooks, W. H., Fraser, R. E., Herman, J. A., Jacobs, R. S., McDonough, J. G., Bonanni, P., Harrison, B., Junot, A., & Kirk, J. (1990). *SIMNET Semi-Automated Forces (Version 3.x): Functional Specification* (Tech. Rep. PTR-4043-15-0200-4/90). Perceptronics.

Jacobs, R. S., McDonough, J. G., & Crooks, W. H. (1990). Semi-Automated Forces in Distributed Simulation. Overheads of talk presented at the DARPA Conference on Behavioral Representation in Semi-Automated Forces, Ft. Knox, KY.

Rosenbloom, P. S. & Newell, A. (1993). Symbolic Architectures: Organization of Intelligence. In T. A. Poggio & D. A. Glaser (Eds.), *Exploring Brain Functions: Models in Neuroscience*. Chichester, England: John Wiley & Sons.

Rosenbloom, P. S., Laird, J. E., Newell, A., & McCarl, R. (1991). A preliminary analysis of the Soar architecture as a basis for general intelligence. *Artificial Intelligence*, *47*, 289-325.

Rosenbloom, P. S., Laird, J. E., & Newell, A. (Eds.). (1993). *The Soar Papers: Research on Integrated Intelligence*. Cambridge, MA: MIT Press.

Stobie, I., Tambe, M., & Rosenbloom, P. S. (1993). Flexible integration of path-planning capabilities. W. J. Wolfe & W. H. Chun (Eds.), *Mobile Robots VII*. Boston, MA, Proceedings SPIE 1831.

Tambe, M., Johnson, W. L., Jones, R. M., Koss, F., Laird, J. E., Rosenbloom, P. S., & Schwamb, K. B. (1995). Intelligent agents for interactive simulation environments. *AI Magazine*, *16*, 15-39.