Introducing the Tileworld: Experimentally Evaluating Agent Architectures

Martha E. Pollack
Artificial Intelligence Center *and*
Center for the Study of Language and Information
SRI International
333 Ravenswood Ave.
Menlo Park, CA 94025

Marc Ringuette
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

We describe a system called Tileworld, which consists of a simulated robot agent and a simulated environment which is both dynamic and unpredictable. Both the agent and the environment are highly parameterized, enabling one to control certain characteristics of each. We can thus experimentally investigate the behavior of various meta-level reasoning strategies by tuning the parameters of the agent, and can assess the success of alternative strategies in different environments by tuning the environmental parameters. Our hypothesis is that the appropriateness of a particular meta-level reasoning strategy will depend in large part upon the characteristics of the environment in which the agent incorporating that strategy is situated. We describe our initial experiments using Tileworld, in which we have been evaluating a version of the meta-level reasoning strategy proposed in earlier work by one of the authors [5].

Topic: Automated Reasoning
Subtopics: Planning and Scheduling, Resource-Bounded Reasoning,
Experimental Evaluation of Planning Systems

[0]†This research was supported by the Office of Naval Research under Contract No. N00014–85–C–0251, by a contract with the Nippon Telegraph and Telephone Corporation and by a gift from the System Development Foundation.

1 Introduction

Recently there has been a surge of interest in systems that are capable of intelligent behavior in dynamic, unpredictable environments. Because agents inevitably have bounded computational resources, their deliberations about what to do take time, and so, in dynamic environments, they run the risk that things will change while they reason. Indeed, things may change in ways that undermine the very assumptions upon which the reasoning is proceeding. The agent may begin a deliberation problem with a particular set of available options, but, in a dynamic environment, new options may arise, and formerly existing options disappear, during the course of the deliberation. An agent that blindly pushes forward with the original deliberation problem, without regard to the amount of time it is taking or the changes meanwhile going on, is not likely to make rational decisions.

One solution that has been proposed eliminates explicit execution-time reasoning by compiling into the agent all decisions about what to do in particular situations [1, 6, 12]. This is an interesting endeavor whose ultimate feasibility remains an open question, but we and others believe that in complex domains, the exclusive use of compilation techniques is impractical [8, 9, 14].

An alternative is to design agents that perform explicit reasoning at execution time, but manage that reasoning by engaging in *meta-level reasoning*. Within the past few years, researchers in AI have provided theoretical analyses of meta-level reasoning, often applying decision-theoretic notions to it [3, 11, 15]. In addition, architectural specifications for agents performing meta-level reasoning have been developed [5], and prototype systems that engage in meta-level reasoning have been implemented [7, 10]. The project we describe in this paper involves the implementation of a system for experimentally evaluating competing theoretical and architectural proposals.

More specifically, we have been constructing a system called Tileworld, which consists of a simulated robot agent and a simulated environment which is both dynamic and unpredictable. Both the agent and the environment are highly parameterized, enabling one to control certain characteristics of each. We can thus experimentally investigate the behavior of various meta-level reasoning strategies by tuning the parameters of the agent, and can assess the success of alternative strategies in different environments by tuning the environmental parameters. Our hypothesis is that the appropriateness of a particular meta-level reasoning strategy will depend in large part upon the characteristics of the environment in which the agent incorporating that strategy is situated. We shall describe below how the parameters of our simulated environment correspond to interesting characteristics of real, dynamic environments.

In our initial experiments using Tileworld, we have been evaluating a version of the meta-level reasoning strategy proposed in earlier work by one of the authors [5]. However, the Tileworld can be used to evaluate a range of competing proposals, such as the ones mentioned above: agents instantiating many alternative proposals can readily be imported into the Tileworld environment.

```
# # # # # # # # # # # # # # # # # # # #
#     T   T         T       T         #
#       #                   2 2     T #
#       #                   2         #
#       # # 5             T           #
#     # # # 5 T                       #
#         # 5   T         a         T #
#       T                             #
#         T     T             T T     #
#               # T # T   T           #
#       T       # # # #               #
#     #           # #         T       #
#     #   T   T   T       T           #
#     #                       #       #
#                         T   #   #   #
#     T                   # # # # #   #
#     # # # # #                       #
#     #                     T T       #
#                   T         T     T #
# # # # # # # # # # # # # # # # # # # #
```

a = *agent*, # = *obstacle*, T = *tile*, < *digits* > = *hole*

Figure 1: A Typical Tileworld Starting State

2 The Tileworld Environment

The Tileworld is a chessboard-like grid on which there are agents, tiles, obstacles, and holes. An agent is a unit square which is able to move up, down, left, or right, one cell at a time. A tile is a unit square which behaves like a tile: it slides, and rows of tiles can be pushed by the agent. An obstacle is a group of grid cells which are immovable. A hole is a group of grid cells, each of which can be "filled in" by a tile when the tile is moved on top of the hole cell; the tile and hole cell disappear, leaving a blank cell. If a hole becomes completely filled, the agent gets points for filling it in. The agent knows ahead of time how valuable the hole is; its overall goal is to get as many points as possible by filling in holes.

A Tileworld simulation takes place dynamically: it begins in a state which is randomly generated by the simulator according to a set of parameters, and changes continually over time. Objects (holes, tiles, and obstacles) appear and disappear at rates determined by parameters set by the experimenter, while at the same time the agent moves around and pushes tiles into holes. The dynamic aspect of a Tileworld simulation distinguishes it from many earlier domains that have been used for studying AI planning, such as blocks-world.

The Tileworld is a rough abstraction of the Robot Delivery Domain, in which a mobile robot roams the halls of an office delivering messages and objects in response to human requests.[1] We have been able to draw a fairly close correspondence between the two domains (i.e., the appearance of a hole corresponds to a request, the hole itself corresponds to a

[1] Various projects at SRI have employed this domain, some of them also employing an actual mobile robot, Flakey.

delivery location, tiles correspond to messages or objects, the agent to the robot, the grid to hallways, and the simulator time to real time).

Features of the domain put a variety of demands on the agent. Its spatial complexity is nontrivial: a simple hill-climbing strategy can have modest success, but when efficient action is needed, more extensive reasoning is necessary. But the time spent in reasoning has an associated cost, both in lost opportunities and in unexpected changes to the world; thus the agent must make tradeoffs between speed and accuracy, and must monitor the execution of its plans to ensure success. Time pressures also become significant as multiple goals vie for the agent's attention.

The Tileworld can be a good test of an agent's abilities to behave intelligently in a dynamic, unpredictable environment. But a single Tileworld simulation, however interesting, will give only one data point in the design space of robot agents. To explore the space more vigorously, we must be able to vary the challenges that the domain presents to the agent. We have therefore parameterized the domain, and provided "knobs" which can be adjusted to set the values of those parameters.

The knob settings control the evolution of a Tileworld simulation. Some of the knobs were alluded to earlier, for instance, those that control the frequency of appearance and disappearance of each object type. Other knobs control the number and average size of each object type. Still other knobs are used to control factors such as the shape of the distribution of scores associated with holes, or the choice between the instantaneous disappearance of a hole and a slow decrease in value (a hard bound versus a soft bound). By adjusting the knobs, one can allow conditions to vary from something resembling an unconstrained football field to something like a crowded maze, or from a fixed puzzle to constantly changing chaos. For each set of parameter settings, an agent can be tested on tens or hundreds of randomly-generated runs automatically. Agents can be compared by running them on the same set of pseudo-random worlds; the simulator is designed to minimize noise and preserve fine distinctions in performance. We will describe the form of an experiment more precisely in a later section.

3 Using Plans To Constrain Future Reasoning

The agent we have implemented and used in our experiments instantiates an architecture for meta-level reasoning presented in [5]. The architecture builds on observations made by Bratman [4] that agents who are situated in dynamic environments benefit from having plans because their plans can constrain the amount of subsequent reasoning they need to perform. Two constraining roles of plans will concern us here [2]:

- An agent's plans focus subsequent means-end reasoning so that the agent can, in general, concentrate on elaborating its existing plans, rather than on computing all possible courses of action that might be undertaken.
- An agent's plans restrict the set of further potential courses of action it needs to give full consideration to, by filtering out options that are inconsistent with the performance of what it already plans to do.

[2] Additional constraining roles have also been postulated [4, 13].

The first role of plans has always been at least implicit in the standard models of AI planning: AI planners compute means to goals that the agent already has. The second has a more dramatic effect on the architecture we are investigating: it leads to the introduction of a *filtering mechanism*, which manages execution time reasoning by restricting deliberation, in general, to options that are compatible with the performance of already intended actions. (To have the desired effect of lessening the amount of reasoning needed, the filtering mechanism must be computationally inexpensive, relative to the cost of deliberation.)

Of course, a rational agent cannot *always* remain committed to its existing plans. Sometimes plans may be subject to reconsideration or abandonment in light of changes in belief. But if an agent constantly reconsiders its plans, they will not limit deliberation in the way they need to. Thus, an agent's plans should be reasonably stable.

To achieve stability while at the same time allowing for reconsideration of plans when necessary, the filtering mechanism should have two components. The first checks a new option for compatibility with the existing plans. The second, an override mechanism, encodes the conditions under which some portion of the existing plans is to be suspended and weighed against some other option. The filter override mechanism operates in parallel with the compatibility filter. For a new option to pass through the filter, it must either pass the compatibility check or else trigger an override by matching one of the conditions in the override mechanism.

An agent's filter override mechanism must be carefully designed to embody the right degree of sensitivity to the problems and opportunities that arise in its environment. If the agent is overly sensitive, willing to reconsider its plans in response to every unanticipated event, then its plans will not serve sufficiently to limit the number of options about which it must deliberate. On the other hand, if the agent is not sensitive enough, it will fail to react to significant deviations from its expectations.

The options that pass through the filter are subject to deliberation. The deliberation process is what actually selects the actions the agent will form intentions towards. In other words, it is the deliberation process that performs the type of decision-making that is the focus of traditional decision theory. The filtering mechanism thus serves to frame particular decision problems, which the deliberation process then solves.

The process of deliberation is different from means-ends reasoning in our view, and this distinction is worth discussing further. As we see it, deliberation is deciding *which* of a set of options to pursue, while means-ends reasoning is more a process of determining *how* to achieve a given goal. We see means-ends reasoning producing *options* (candidate plans to achieve a goal), which can then be the subject of deliberation.

This may be a surprising distinction to those familiar with the standard AI planning paradigm, in which the job of a planner is usually to produce the single best plan according to some set of criteria. Any deliberation which is to be done in such a system is done by the planner, and it could be argued that a planner is the best place for such reasoning. Certainly some pruning of alternatives must be done in the planner; however, there are reasons to believe that some deliberation belongs outside the planner. In some situations it is appropriate to have several means-ends reasoners with differences in solution quality and time required; these must be invoked appropriately and a single solution chosen. In other circumstances it is desirable to engage in a decision-theoretic analysis of competing alternatives. Consequently, we have maintained the distinction between deliberation and means-ends reasoning in our system.

4 The Tileworld Agent

The Tileworld agent was constructed to test some of the ideas discussed in the previous section. We therefore made some strong commitments in our design: the agent will *make plans*; it will maintain an *intention structure*, represented as a time-ordered set of tree-structured plans, to which the agent is fairly strongly committed; it will periodically reason about the status of its intentions; and this reasoning process will include the *filtering* mechanism discussed above.

We are interested in situations in which an agent has enough time to conduct significant reasoning, but little enough time that the cost of reasoning must be taken into account. In such situations, it is highly advantageous for the agent to be able to engage in reasoning while carrying out an action previously decided upon. We have therefore chosen to allow our agent to simultaneously reason about what to do, and perform actions and perceive changes in its environment. Our model is of a robot with two sets of processing hardware. One processor executes a short control cycle (the *act cycle*), acting on previously formulated plans and monitoring the world for changes. The second processor executes a longer cycle (the *reasoning cycle*) which permits computations with lengths of up to several seconds. Although this model incurs a certain cost in the complexity of synchronizing the two processes, it allows for a balance of computational flexibility and reactivity. We feel that this is a sensible choice for the design of a real mobile robot, but in our current system we simulate the concurrency for the sake of convenience.

The act cycle is straightforward; the agent performs those acts that have been identified during the previous reasoning cycle, monitoring for limited kinds of failures. Perception also occurs during the act cycle: the agent can access a global map of the world that indicates the locations of all objects, as well as the score and time remaining to timeout for all holes.[3]

The reasoning cycle makes decisions about what goals to pursue and how to pursue them. The portion of the agent architecture that controls reasoning is depicted in Figure 2. As illustrated there, new options for consideration can come from two sources. First, the agent may perceive environmental changes that suggest new options—in Tileworld, this occurs when new holes or tiles appear. Second, options may be suggested by a means-end reasoner, the bulk of which is a special-purpose route planner. The means-end reasoner suggests plans that can serve as means to already intended ends. For example, it may suggest moving to a certain location in order to push a particular tile into some hole, when the filling of that hole is a component of the agent's current *intention structure*.

Options from both sources are then theoretically subject to filtering. We have so far confined filtering to top-level options, i.e., options to fill a particular hole. However, at least some extensions to subordinate options are obvious: for example, the use of a particular tile to fill one hole should be filtered as incompatible when there already exists a plan to use that tile for a different hole.

Recall that the filtering mechanism must determine whether an option is compatible with the agent's existing plans. If it is incompatible, it must also determine whether it triggers an

[3] We plan in the future to investigate more-local perception strategies.

```
                                            Environment
                          ----------             |
                         | Means-   |            V
                    ---> | Ends     |       ------------
                    |    | Reasoner |      | Perception |
                    |     ----------        ------------
                    |         |                  |
                    |         V                  V
                    |       ---------------------
                    |      | Filtering Mechanism |
             -----------    ---------------------
            | Intention |---------       |
Action <----| Structure |        |       |
             -----------         V       V
                    ^          ----------------------
                    |         | Deliberation Process |
                    |          ----------------------
                    |_________________|
```

Figure 2: Tileworld Agent Architecture

override, that is, whether it is potentially important enough that the agent should nonetheless consider it.

Compatibility checking of top-level options as implemented to date is very simplistic. If the agent has a current intention to fill a particular hole (say, hole N) right now, then it is the case that filling any other hole M right now will be incompatible with the existing intention. The option to fill hole M now will *not* survive the compatibility filter. Thus, deliberation about whether to abandon work on N, and instead work on M, will depend upon the override mechanism.

In the simplest version of the override mechanism, a threshold level is set to some constant v, which represents the increase in score value that the new hole must have over the old one to be worthy of further consideration. Recall that triggering an override will not necessarily result in the agent's abandoning its currently executing plan; that depends upon the details of the deliberation component, described below. However, if $(score(M) - score(N)) <= v$, then the agent will not even consider abandoning its filling of N, and will defer without further consideration attempts to fill M. Notice that if we set the threshold value to $-\infty$, deliberation will occur whenever the environment changes in such a way as to provide a new potential option. It is useful to introduce some terminology from Bratman *et al.*: when an agent is very sensitive to its environment, willing to reconsider its plans in response to a wide range of events, we say that it is *cautious*. If it tends instead to "stick to its guns" regardless of what is happening in the surrounding environment, we say that it is *bold*. For a Tileworld agent, the lower the threshold value, the more cautious it is.

Options that survive the filtering mechanism are then subject to deliberation. In the example we have been describing, assume that filling M now survives the filtering mechanism. Then it is necessary to deliberate about whether in fact to adopt that intention, and begin work on filling M, or whether to continue with the current plan of filling N. Alternative deliberation strategies can be chosen in Tileworld by the setting of a parameter. We currently have implemented two deliberation modules.

The simpler deliberation module evaluates competing top-level options by selecting the one with the higher potential score. Thus, when the threshold parameter for the override mechanism is nonnegative, this mode of deliberation will always select the new competing option over the one that was previously held. With negative threshold values, it will always select the already executing action. This illustrates a general point: if deliberation is extremely simple, it may be redundant to posit separate deliberation and filtering processes.

A slightly more sophisticated deliberation strategy estimates the subjective expected utility (SEU) of a top-level goal. For a given option to fill a hole M, SEU is estimated as a function of $score(M)$, time available to fill M, distances of the agent and available tiles from M, and the size of M; these factors can be combined into an improved measure of the likelihood of success of filling M in the time allotted. More specifically, our current SEU-estimator function is:

$$SEU(h) = \frac{score(h)}{dist(a,h) + tileavail(h)}$$

The function $dist(x,y)$ represents the distance between two locations on the grid: in this case, a is the agent, and h the hole. $Tileavail(h)$ is defined as:

$$tileavail(h) = \sum_{i=1}^{n} 2 * dist(t_i, h)$$

where t_i is the i^{th} nearest hole to h, and n is the size of h. Informally, $tileavail(h)$ represents the total distance the agent will have to travel, once it is at or near h, to complete its task of filling the hole. The distance to each of the nearest n tiles is multiplied by 2 because the agent must make a round trip, traveling to each tile to reach it, and then pushing it back to the hole. Our current SEU-estimator does not take into account the amount of time left until the hole times out.

With this more sophisticated mode of deliberation, the agent may decide to continue with its current plan to fill N, even if filling M has a higher potential score; this will occur if the filling of N is expected to take significantly less time.

We intend to design additional deliberation modules, including one that simulates complete means-end reasoning for options under consideration.

5 Experiments With Our Agent

With both the simulator and the agent in place, we are in a position to conduct experimental studies of the the performance of the agent. By adjusting the Tileworld "knobs", we can control a number of domain characteristics. We can vary what we call *dynamism* (the rate at which new holes appear), *hostility* (the rate at which obstacles appear), *variability of utility* (differences in hole scores), *variability of difficulty* (differences in hole sizes and distances from tiles), and *hard/soft bounds* (holes having either hard timeouts or gradually decaying in value). There are also variables we can adjust in the agent: *act/think rate* (the relative speeds of acting and thinking), the filter's *threshold level*, and the *sophistication of the deliberation mechanism*.

Experiment 1

To begin with, we set all of these parameters to provide a baseline environment which is dynamic, variable, and moderately paced. In this environment, a competent agent can achieve reasonable scores, but is penalized for wasting time or making poor choices. We will start by comparing the simple deliberation mechanism, based on score value, with the SEU evaluator, which provides a better estimate of marginal utility. For orientation, we have also included the results of a human playing the role of the agent in the same simulation; and to gain an idea of the benefit of acting in parallel with reasoning, we have included results for an agent that that acts and reasons serially.

All of these agents were tested in the baseline environment and in a similar, but more rapidly changing one. In the faster environment, objects appear and disappear on the average ten times more quickly, but the agent can also move ten times more quickly. However, the agent's reasoning takes place at the same rate of speed as in the baseline case, so the opportunity cost of reasoning is correspondingly greater in the faster environment. The agents were all evaluated by taking the average score from 30 trials; the human performed 10. Each trial is a self-contained simulation with a duration of 5000 ticks of the clock.[4]

Experiment 1

Agent	SEU	SEU/serial	Simple	Simple/serial	Human
normal speed	396	353	347	291	468
10x faster	256	234	183	152	3

The differences here are quite apparent. In the normal speed environment, the human subject performed best. This resulted from his having more-sophisticated planning capabilities than the robot agent. But in the faster environment, the human's planning "tricks" were insufficient, and he could not keep up with the pace of change.

The robot agents were better able to adjust to the more rapidly changing environments, but it is clear that the cost of reasoning is still significant for them. This is evident both from an overall decrease in score in the high-speed environment, and from the superiority of the robot agents that could reason and act in parallel.

The other distinction of note is that the SEU evaluator performs better than the simple evaluator, as we might expect.

Experiment 2

We now move on to our early results in a continuing series of experiments directed at understanding some of the design tradeoffs in our agent. The use of Tileworld to experimentally evaluate our agent architecture is an ongoing project, and we describe these initial results primarily to point out emerging trends. We stress that the hypotheses presented below are

[4]The agent can move once per clock tick.

preliminary; significantly more experimentation and statistical analysis of the results needs to take place before we can make strong claims about the relative appropriateness of any particular agent-design strategy.

In Experiment 2, we attempt to test the usefulness of the filtering mechanism in our agent as implemented, using the SEU evaluator as the deliberation component, and using the most quickly computed evaluation metric, thresholding on the score value, as the filter override mechanism. We vary the threshold from -100 to 100. Since the score for each hole ranges from 1 to 100, a threshold setting of -100 means that every new option is subject to deliberation (a strategy of extreme *caution*), while a setting of 100 means that no new option will ever be considered until the currently executing plan is complete (extreme *boldness*). The resulting scores are summarized in the following table, with each number again representing an average over 30 trials.

Experiment 2

Threshold	-100	-75	-50	-25	0	25	50	75	100
100x slower	434	434	433	427	422	400	405	398	398
normal speed	396	413	393	409	404	398	388	381	371
10x faster	256	265	264	241	251	233	255	251	266

At the slowest speed setting, 100 times slower than our "normal" setting, it is better to do no filtering at all (extreme caution). The scores achieved at this speed decrease monotonically as the threshold is increased. At the normal speed setting, the effect of increased filtering still appears to be negative, but less markedly so. At the fast setting, there seems to be little correlation between threshold level and performance, although the uncertainty in the results, which appears to be in the range of 10-20 points, prevents a sure determination. We hope, in the future, to be able to make even these relatively subtle determinations; the noise in the data comes, we believe, largely from our decision to use actual CPU-time measurements to determine reasoning time. If we wish to get the cleanest trials possible, we may need to use a time estimate that does not depend on the vagaries of the underlying machine and Lisp system. Failing that, we will need to model the uncertainty involved, and run larger trial sets.

To sum up the results of this experiment, we see that filtering is harmful at slow speeds, and even at high speeds does not give a net benefit. Our hypothesis is that the time cost of the SEU evaluator is not very high, and consequently, it is usually worth taking the time to engage in extra deliberation about new opportunities. The fact that filtering is less detrimental in the faster environment leads us to hypothesize that there may be a break-even point at even faster speeds, above which filtering is useful; we intend to test for such a point. We also intend to implement more accurate (and costly) deliberation mechanisms in the near future. For these, filtering may be much more valuable; perhaps the SEU-estimator is efficient enough that it can itself be used as the filter override mechanism for the more complex deliberation components.

Experiment 3

We speculated that the SEU-estimator, as described in Section 4, may be deficient in an important way: it does not consider the time cost of means-end reasoning already performed.

Consequently, in our third experiment, we modified the deliberation functions by adding a bias in favor of existing intentions, since typically at deliberation time, some means-end reasoning about how to achieve these has already taken place. This is distinct from Experiment 2, in which we adjusted the *filtering* mechanism in an attempt to save deliberation time; here we investigate a bias in the deliberation process itself with the intent of reducing the time cost of *means-end reasoning.*

We consider two cases. In the first, deliberation is done by the simple evaluator, and we apply a bias towards existing intentions equal to a fixed number of points. In the second, deliberation is done by the SEU evaluator, and we apply a bias equal to a fraction of the current SEU. Thus, for example, with a 100% bias, a newly appearing hole must have double the SEU of the current one to be adopted as a new intention. The environment settings and simulation sizes are the same as for Experiment 2.

Experiment 3

SEU Evaluation					
Bias (percent)	0	25	50	100	200
100x slower	434	427	412	423	417
normal speed	404	385	399	392	394
10x faster	259	243	259	252	250
Simple Evaluation					
Bias (points)	0	25	50	75	100
100x slower	352	354	355	351	354
normal speed	347	372	369	363	359
10x faster	183	187	174	203	192

As shown by the experimental results, bias in the deliberator does not appear to have a clear effect on total performance. For the simple evaluator this isn't terribly surprising; it provides a fairly weak assessment of a hole's actual potential value in any case. We expected to see much more effect of bias on the SEU evaluator, however. Two hypotheses are available to explain this. First, our test environment may have too many opportunities available, minimizing the potential cost of high bias: if the agent spends most of its time doing something with high utility, a few missed opportunities will not have a significant impact on the final score. This hypothesis can be tested in a less favorable environment. Second, it may be the case that means-end reasoning in the current implementation is too inexpensive, minimizing the potential benefit of high bias. This hypothesis can be tested by increasing the *size* of the environment to increase the planning time required; the addition of more complex planning routines would also provide situations in which there is a higher time cost associated with planning.

6 Conclusion

The experiments we have run to date have included some important milestones in the Tileworld effort. The Tileworld domain has been demonstrated, and has been shown to be a viable system for evaluating agent architectures. The Tileworld agent was demonstrated and used to test differing deliberation and filtering strategies as described in [5].

We continue to investigate the question of how an agent should structure and control its computational effort. We believe that the architecture discussed here is a special case of a more general framework, and we are working towards a definition of that framework and its verification in our domain. We also see the Tileworld testbed as a good basis for comparison of other agent architectures proposed in the literature, and we strongly encourage other researchers to demonstrate their agents in our domain.[5]

The goal of our experiments is an improved understanding of the relation between agent design and environmental factors. In the future, when faced with a performance domain for an agent, one should be able to draw on such an understanding to choose more wisely from the wide range of implementation possibilities available.

[5]The Tileworld domain is relatively clean and portable, is written in CommonLisp, and is available electronically over the Internet from Marc Ringuette by sending electronic mail to `mnr@cs.cmu.edu`.

References

[1] P. E. Agre and D. Chapman. Pengi: An implementation of a theory of activity. In *AAAI-87, Proceedings of the National Conference on Artificial Intelligence*, Seattle, Wa., 1987.

[2] J. Blythe and T. M. Mitchell. On Becoming Reactive. In *Proceedings of the 6th International Workshop on Machine Learning*, Cornell University, June 1989.

[3] M. Boddy and T. Dean. Solving time-dependent planning problems. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Detroit, MI, 1989.

[4] M. E. Bratman. *Intention, Plans and Practical Reason.* Harvard University Press, Cambridge, Ma., 1987.

[5] M. E. Bratman, D. J. Israel, and M. E. Pollack. Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4:349-355, 1988.

[6] R. A. Brooks. Planning is just a way of avoiding figuring out what to do next. Technical Report 303, MIT, 1987.

[7] P. R. Cohen, M. L Greenberg, D. M. Hart, and A. E. Howe. Real-time problem solving in the Phoenix environment. In *Proceedings of the Workshop on Real-Time Artificial Intelligence Problems*, Detroit, MI, 1989.

[8] B. D'Ambrosio and M. Fehling. Resource bounded-agents in an uncertain world. In *Proceedings of the AAAI Symposium on Limited Rationality*, Stanford, Ca., 1989.

[9] J. Doyle. Artificial intelligence and rational self-government. Technical Report CS-88-124, Carnegie Mellon University, Pittsburgh, Pa., 1988.

[10] M.P. Georgeff and F.F. Ingrand. Decision-making in an embedded reasoning system. In *Proceedings of the International Joint Conference on Artificial Intelligence*, Detroit, Mi., 1989.

[11] E. J. Horvitz. Reasoning about beliefs and actions under computational resource constraints. In *Proceedings of the 1987 Workshop on Uncertainty in Artificial Intelligence*, Seattle, WA, 1987.

[12] L. P. Kaelbling. Goals as parallel program specifications. In *AAAI-88, Proceedings of the Seventh National Conference on Artificial Intelligence*, Saint Paul, Minnesota, 1988.

[13] M. E. Pollack. Overloaded expectations. In preparation, 1990.

[14] J.L. Pollock. Oscar: A general theory of rationality. In *Proceedings of the AAAI Symposium on Limited Rationality*, Stanford, Ca., 1989.

[15] S. J. Russell and E. H. Wefald. Principles of metareasoning. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, Toronto, 1989.