IJCAI 2005 Workshop on Reasoning, Representation, and Learning in Computer Games

Interfacing the D'Artagnan Cognitive Architecture to the Urban Terror First-Person Shooter Game

Bharat Kondeti, Maheswar Nallacharu, Michael Youngblood and Lawrence Holder
University of Texas at Arlington
Department of Computer Science and Engineering
Box 19015, Arlington, TX 76019
{kondetibharat,mailbackmahesh}@yahoo.com, {youngbld,holder}@cse.uta.edu

Abstract

The D'Artagnan Cognitive Architecture (DCA) is a multi-agent framework that supports the study of attention as a means to realize intelligent behavior by weighting the influence of different agents as they collectively determine the next action. We have interfaced the DCA to the Urban-Terror (UrT) first-person shooter game and defined several worlds of increasing complexity in order to test the DCA's ability to perform well in these worlds and demonstrate the usefulness of shifting attention among different agents. We have implemented several reflex agents and a path planner to help DCA play the UrT game. Experimental results indicate that a DCA-based player using a combination of action-determining agents can be successful when no single agent can complete the task.

1 Introduction

Interactive computer games have been considered human-level AI's "killer app" [Laird and van Lent, 2001] in that current games have a sufficient level of realism to require human-level intelligence to play well. Laird and van Lent's work along these lines with the SOAR cognitive architecture and the Unreal Tournament game explored the current limits of AI to play these games [Laird, 2002]. Motivated from this challenge, but with an alternative view of the design of cognitive architectures, we have begun development on the D'Artagnan Cognitive Architecture (DCA) and an interface between it and the Urban Terror (UrT) first-person shooter game. The DCA is a novel approach to cognitive architectures based on a Minskian society-of-agents approach [Minsky, 1988] of psychologically-inspired, possibly competing, agents with a global focus-of-attention influence over the agents to achieve robust, human-consistent intelligent behavior. In previous work we have presented metrics for human-consistency and comparison of human and DCA behaviors [Youngblood and Holder, 2003].

In this paper we describe the DCA, the UrT game, and the interface between the two. Fundamental to a player's performance in such environments is the ability to reason spatially. Therefore, we have also implemented a path planner based on the work of [Hill, 2002] that generates a topological graph from the UrT world maps and uses the graph to find paths between the agent's starting and goal location. While there is a large body of work in representations and algorithms for path planning in many domains (e.g., see O'Neill's [2004] work on a mesh representation for game worlds to support path planning), our work is unique in its ability to automatically generate a topological graph from the UrT's map representation, which is the means by which different UrT world scenarios are distributed to gamers.

To test the DCA approach, we define multiple tasks in five different UrT maps and evaluate the performance of a reflex-agent-based DCA while playing UrT. Our goal is to evaluate the hypothesis that the DCA consisting of multiple action-generating agents, controlled by a global attention agent, can accomplish tasks too difficult for a single-agent-based DCA. This hypothesis is similar to that confirmed in Reynolds' [1999] work, where he exhibited human-consistent steering behavior using a linear combination of numeric values from lower-level behaviors (e.g., flee danger, avoid obstacles). However, the DCA must choose among a discrete set of actions for which we propose an approach based on an adaptive focus of attention.

2 D'Artagnan Cognitive Architecture (DCA)

The D'Artagnan Cognitive Architecture (DCA, http://ailab.uta.edu/dca) [Youngblood, 2000; Youngblood and Holder, 2003] is based on the work of existing cognitive architectures, robotics research, and human-consistent cognitive models, centered on a specific task. DCA consists of twelve components, or models of cognition (see Figure 1). The twelve DCA models consist of the action model, action evaluation model, attention model, decision model, effectual model, emotion model, goal model, learning model, learning evaluation model, memory model, perceptual model, and reflex model. Models are implemented using one or more agents in a multi-agent framework, e.g., several different types of learning techniques may be underlying the learning model, and all compete for the bandwidth to influence DCA's behavior. When connected, these components form an architectural system for learning and adapting to an environment. Figure 1 depicts one possible set of connections between models, but in reality the models are completely connected. The communication bandwidth across a connec-

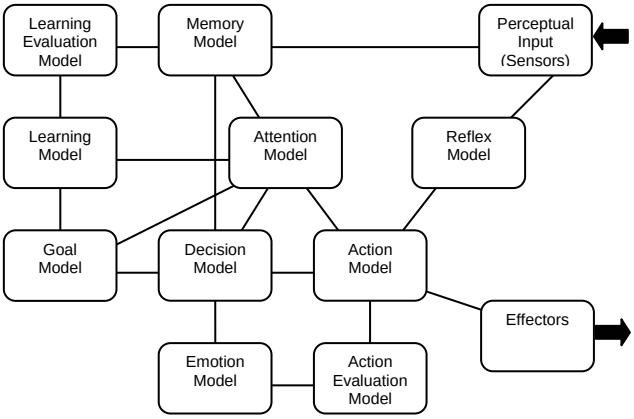

Figure 1. The D'Artagnan Cognitive Architecture (DCA).

tion varies dynamically, as controlled by the attention model.

When percepts are received and stored into memory, the appropriate models are triggered and start processing. Some models will select goals and propose actions based on learning from past actions, while other models will use deliberative planning to determine the next action leading toward a goal. The proposed actions are available at any time for the action model, which selects the action to take. The selected action is executed by the effectors.

Selection among the set of possible actions is affected by a number of factors. The learning model not only generates knowledge and possible actions, but also evaluates past decisions to learn the action's effectiveness for a particular goal. The emotion model provides a suppressor or enabler signal of an action based on environmental situations. Strong emotions can completely inhibit certain actions and enable others (e.g., fear will inhibit charging an opponent and enable retreat). The final dispatcher of influence is the attention model, which controls the weights of all edges and thus controls communication between models and the confidence level of possible decisions generated by these models. The attention model also controls the timing of decisions, following the anytime paradigm, to produce the best possible decision at given time intervals. Based on the human ability to focus the mind on different thought mechanisms for different situations, the attention model can stop and restart other models to enforce desired behavior.

3 Urban Terror (UrT)

We have begun development on interfacing DCA to a visually and tactically realistic urban warfare simulator called Urban Terror (UrT, http://www.urbanterror.net), which is built on top of the cross-platform (Win32, Linux, Mac OS X) Quake III Arena game engine. UrT is a first-person shooter (FPS) game developed by Silicon Ice Development. At present UrT is offered as an entertainment-based game, and to our knowledge, has not been deployed for any other commercial or military use. As part of this project we have

Figure 2. Urban area map (left) used by the DCA-UrT project and a screen shot of the game (right).

implemented an interface to the UrT game that allows the DCA (or any other system) to extract perceptual information from the UrT game and perform actions in UrT. UrT supports several challenging world maps (e.g., Figure 2 depicts an UrT urban map and a game screenshot) and game scenarios (e.g., capture the flag, bomb-defuse, and free for all). We have also defined our own simplified games within these worlds that still portray realistic urban warfare scenarios. We have developed mechanisms for logging game information in XML in order to extract a player's behavior. Eventually, we plan to have human players play our worlds in order to capture their play, which will serve as part of an evaluation metric for an agent's consistency with human behavior.

4 DCA-UrT Interface

The DCA and UrT are interfaced via shared memory to exchange percepts and actions. The shared memory is used to read and write percepts and actions with lower communication latency and lower computational burden on the game engine. A visual interface called UrTInterface (see Figure 3) has been developed for UrT to display all the percept information that can be obtained from the game and also acts as a virtual keyboard to play the game. This section describes the main aspects of interfacing DCA to UrT. For more interface details, see [Kondeti, 2005].

4.1 Modifications to UrbanTerror

Since UrbanTerror (UrT) is a realistic shooter game, with sophisticated worlds and adversaries, and DCA is still in its infancy, a number of modifications had to be done to UrT before DCA can play it. The main concern of DCA is to navigate towards the goal while avoiding obstacles. So the opponents in the game had to be removed since their goal is to kill the player and end the game. This is done by deliberately not allowing the game to start any opponents in the game, giving DCA more time to successfully navigate to the goal without getting killed by opponents.

The goal of the DCA is to get the opponent's flag. The rules for "Capture the Flag" mode in UrT require the player

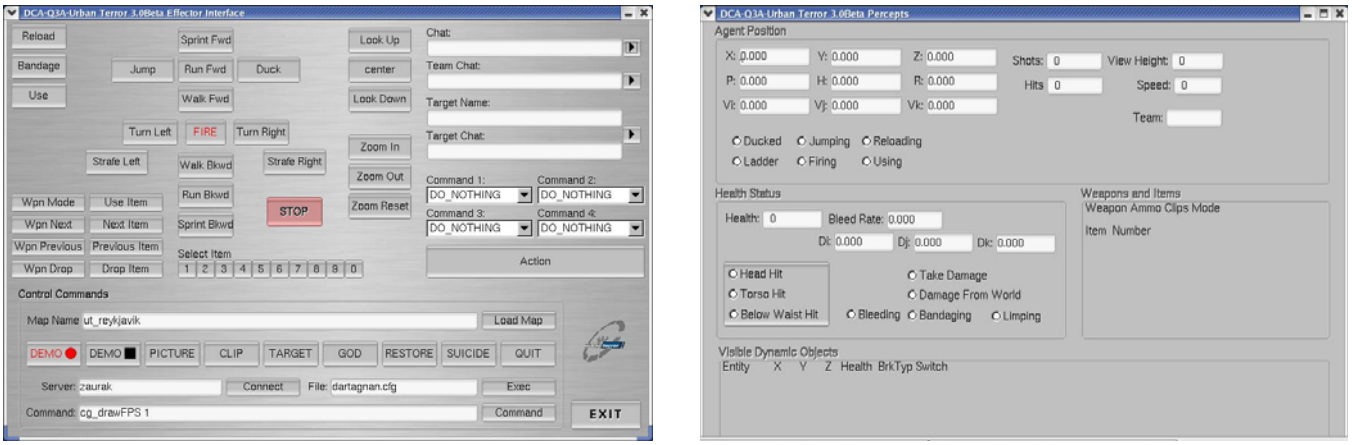

Figure 3. DCA-UrT effector interface (left) and percept interface (right).

to get to the opponent's flag and return back to his base without being killed or losing the flag. Since there are no opponents and the task for DCA to navigate toward the goal is itself very difficult, the complexity of the game is minimized by finishing the game once the DCA player gets the opponent flag, avoiding the burden of navigating back to the starting location with the flag.

The game is also modified to log information about the time, step, action, player health and player location, which is sufficient to determine if the DCA was able to finish the task, and if so, the time and number of actions taken to reach the goal.

4.2 Modifications to DCA

DCA is modified to handle the percepts obtained from UrT, consisting of 33 different percepts related to the player, 11 different percepts about entities which include opponents if they are present and all the different dynamic objects, and 4 different percepts about weapons. DCA can choose from 29 different actions that can be sent to the game.

A portion of DCA was implemented to produce a reflex agent based cognitive architecture to test the working of DCA with UrT. This includes a perceptual agent for getting information from the UrT environment, a basic memory agent, an attention agent, several reflex agents, a bread-crumb agent, a path-planning agent, an action agent and an effector agent to send actions back to UrT. The implemented portion of the DCA model for UrT is shown in Figure 4 along with the main communication links between the agents.

Only the percept agent and effector agent are interfaced to the local shared memory. All the information that is read by the percept agent from shared memory is first sent to the memory agent. The memory agent creates a data structure of all the information and places the data in the network for the reflex and path-planning agents to consume. All the reflex agents (described in the next section) process this information and send an action to the action agent. The path-planning agent determines waypoints for the BreadCrumb agent. The action agent receives actions from different reflex agents and selects an action based on a policy. Since there are no agents to evaluate the action taken, the policy is hard-coded into the action agent. Each link from a reflex agent is given a weight by the attention agent, and the action is chosen according to this weight. The action thus determined is sent to the effector agent.

4.3 Reflex Agents

There are four reflex agents implemented within the DCA-UrT interface. Some reflex agents maintain a small amount of state information to determine whether or not they are stuck at the same position for a while. The Random Agent depicted in Figure 4 represents one of two random reflex agents: with goal information and without goal information. For the random reflex agent with no goal location the agent randomly chooses one of the navigation actions. For the random reflex agent with goal location the agent always tries to move in a straight path towards the goal. If there are any obstacles in between, and the agent determines that it is struck at the obstacle, then the agent randomly chooses one of several evasive sequences of steps and sends them all sequentially as its actions, in order to move away from the obstacle.

The Ping Agent on the other hand knows the distance to the nearest obstacle in its line of sight. If this distance becomes less than a threshold value, the Ping Agent takes one of the sequences of random steps and continuously takes these sequences of steps until the distance to the nearest obstacle is greater than the threshold value.

For the BreadCrumb agent obstacle-free subgoals are provided which when followed take the agent to the final goal. These subgoals are chosen such that a minimum of them are required to reach the goal. These subgoals are visually identified and hard-coded into the agent, or generated by the path planning agent (see next section). Even though subgoals provided for the bread crumb agent are obstacle free, there is a possibility that the agent may get stuck at an edge of an obstacle or there might be a small obstacle in between subgoals. To accomplish the goals in such scenarios the BreadCrumb agent is played along with

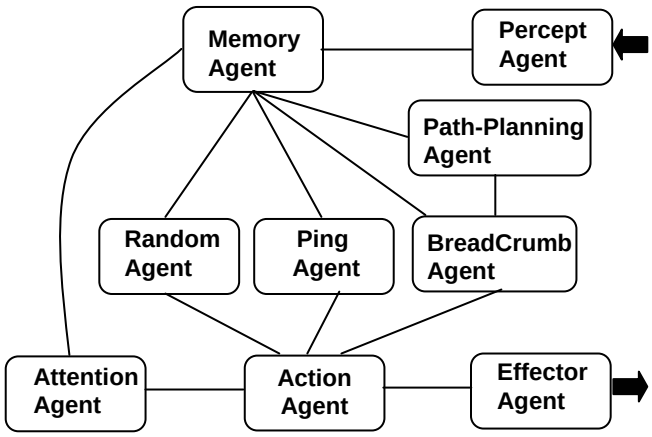

Figure 5. DCA model for UrT interface.

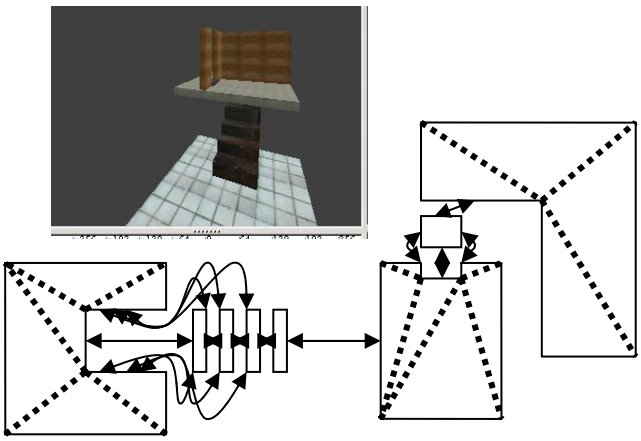

Figure 4. Sample map (above) and its topological graph (below).

one of the random agents. The attention agent keeps track of whether the DCA player is stuck and directs the action agent to place additional weight on the actions from the random agent to get itself free from the obstacle.

4.4 Path Planning Agent

The ability to navigate is a fundamental component of intelligent behavior in real-time simulated worlds. We have developed a path planning agent for the DCA-UrT interface to support the action-generating agents in their quest toward the goal. Currently, the path-planning agent supplies bread crumbs for the BreadCrumb agent to follow. The path planning system is developed in two phases. The first phase converts the UrT game world map representation into a topological graph (TOG) representation of the game world. This is a complex process, since the world map representation was not designed to be used for path planning. The second phase is the one used in the path-planning agent, i.e., it involves searching the TOG to extract path information from the graph structure. We provide an overview of the process here. For more information, see [Nallacharu, 2005].

The first phase of the path planning process consists of the following four tasks. For reference, Figure 5 shows a small world consisting of two floors connected by stairs with a wall and door on the second floor. The bottom of Figure 5 shows the final topological graph for this world.

1. *Parsing the .map file*. The Quake III family of games, of which UrT is one, describes the worlds using a .map file format. The first step is to parse this file to extract the planes, or "brushes", used to describe the world. For example, the world shown in Figure 2 (known as Reykjavík) consists of 1814 brushes.
2. *Find related brushes*. In this step we calculate the relationships (i.e., intersections) between brushes.
3. *Find brush reachabilities*. Reachability information helps decide if one can traverse from a given brush to another. The reachability represents a directed edge connecting two plane surfaces in three-dimensional space. Reachability information also includes a cost metric, which is directly proportional to the inclination and the height values of the brush planes and represents the physical difficulty in traversing between the two brushes.
4. *Generate topological graph*. The process of generating the topological graph starts at a brush and expands along reachability edges in a breadth first manner.

The bottom of Figure 5 shows the TOG for the top world. Notice that each step of the staircase is a node in the graph and the bottom three steps can all be reached directly (perhaps by jumping) from the first floor.

Given the TOG and a starting and goal location, the path-planning agent can generate a set of points along a path between the two locations. We first identify the nodes containing the desired locations by searching the graph using bounding box information. We then find the shortest path between the start and goal nodes based on the edge costs (recall that some edges cost more because they represent more complicated moves for the agent).

The next step is to determine exactly how to move based on the traversing edge in the TOG. Since the nodes represent concave surfaces, simply moving between midpoints of the common edges may not yield a successful path. So, we use a concave polygon triangulation method to divide up the nodes into convex triangles and augment the path at each edge transition with the result of a sub-search through the midpoints of each surface's triangles. The graph at the bottom of Figure 5 shows the triangulation of the node surfaces.

Finally, the points in the path are communicated to the BreadCrumb agent, which translates them into actions. These actions are sent sequentially to the action model.

5 Experimental Results

Two main objectives of the DCA are to show human consistency in its behavior and that a collection of possibly competing models can sometimes accomplish tasks not possible by a single model approach. We have evaluated the human-consistency of the DCA in earlier work using the Quake II game [Youngblood and Holder, 2003]. In the Quake II experiments we defined 100 levels of increasing complexity starting from a single room with the objective right in front

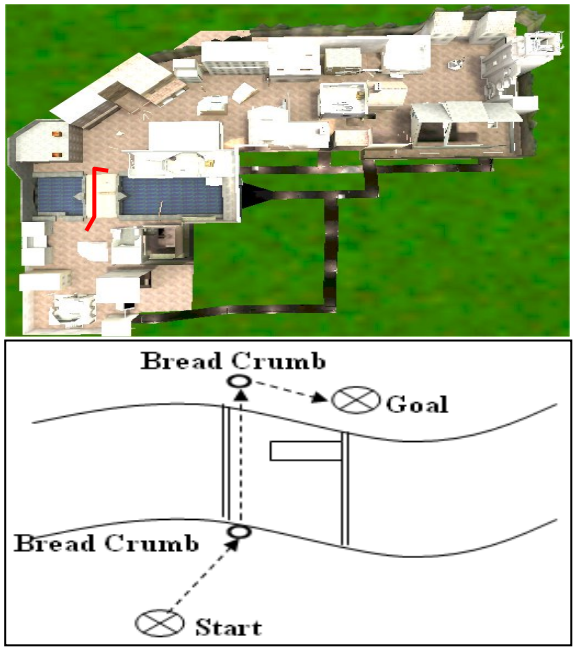

Figure 6. Rommel map (top) and the task 4 scenario of crossing the canal bridge (bottom).

of the agent to a complicated world with multiple adversaries. In addition to having the DCA play these levels, we also collected game play data from 24 different human players. We found that the human players clustered into three groups: novice, intermediate and expert. And we found that the DCA player was consistent with the novice-level human play according to a path edit distance based metric. The Urban Terror game provides a much more realistic urban-warfare scenario based on the Quake III Arena game engine.

The experiments reported here are designed to test the hypothesis that multiple action-generating agents working simultaneously within DCA may yield better game-playing performance than a single agent approach. The agents utilized are the two random agents (with and without goal location), the ping agent, the bread crumb agent with hand-crafted crumb locations, and the bread crumb agent with crumb locations provided by the path-planning agent.

5.1 Test Maps and Tasks

For our experiments we are using the following five different maps included with UrT.

1. *Reykjavik*. This map represents an urban warfare scenario and consists of four large buildings with three floors each separated by many path ways. This map is pictured in Figure 2.
2. *Rommel*. This map (see top of Figure 6) represents a village scenario where most of the buildings are destroyed. A canal flows through the village and there are many obstacles in the form of rubble.
3. *Docks*. This map depicts a warehouse scenario at the edge of a shallow body of water. This map is the most complex of all the five maps with many buildings and rooms that are interconnected in a complex manner.
4. *Riyadh*. This map portrays a desert scenario and consists of two market places far away from each other. Bezier curves are used to construct all the curved surfaces to give a desert effect.
5. *Twinlakes*. This map is built upon mountains covered with snow. The map is completely covered with trees that are climbable. The map also has two houses at either end of the map with a small pond for each house. For this map also Bezier curves were used to construct the mountainous terrain.

For each of the five maps we defined five tasks consisting of capture-the-flag games of increasing difficulty. Table 1 describes the 25 tasks.

5.2 Results

We evaluated the performance of DCA with various reflex agents playing individually and then two or more reflex agents playing together. The metric information used for evaluation is whether the individual reflex agents or cooperative reflex agents were able to accomplish the given task, and if so, the time taken to accomplish the task and the number of actions sent to UrT to finish the task. Each version of the DCA was allowed to play each task three times and the average of the three plays is taken for evaluation purposes.

Figure 7 shows the time take by each of the three single-agent DCAs on the task 3 scenario for the five maps. Except for Rommel, the performance of the bread crumb agent is better than all other agents, and the performance of the ping

Table 1. Five tasks for each of the five DCA-UrT maps (obstacles are between start and goal locations).

	Reykjavik	**Rommel**	**Docks**	**Riyadh**	**Twinlakes**
1	Obstacle-free traversal	Obstacle-free traversal	Obstacle-free traversal	Obstacle-free traversal	Obstacle-free traversal; mountainous terrain
2	One large obstacle	L-shaped obstacle	Two adjoining obstacles with deadend	One small obstacle	One octagonal slippery-surfaced obstacle
3	Start enclosed by wall with door	Cluster of walls	Two clusters of obstacles far from each other	Platform obstacle	Row of four obstacles
4	Traverse large L-shaped alley	Cross bridge over canal (see fig. 6)	Cross narrow bridge to floating dock	Traverse around marketplace	Traverse thru tunnel
5	Obstacle-free staircase traversal	Long traversal, many obstacles	Climb stairs to second floor	Descend ladder to first floor	One room to another through narrow door

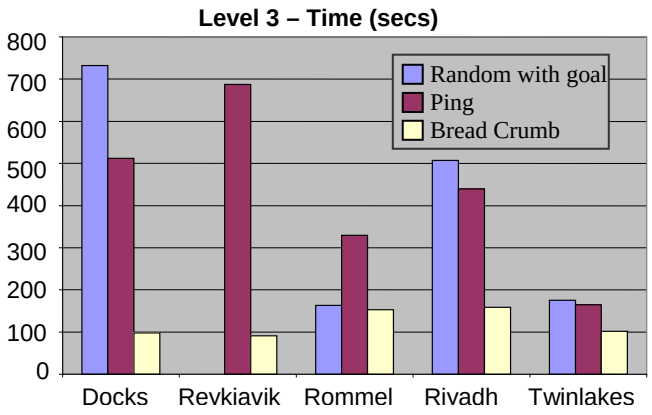

Figure 7. Average time taken by three agents in the task 3 scenario of the five maps.

agent is better than that of the random agent with goal information. For Reykjavik the random agent with goal information could not finish the given task, so the values for that agent are not plotted. For Rommel the performance of the random agent with goal information is better than that of the ping agent and almost equivalent to that of the bread crumb agent. This is because the obstacle present is in the form of a slanted wall, and the random agent simply slides along the wall to reach the goal.

Results for tasks 1 and 2 follow the trend of the bread crumb agent taking the least time, followed by the ping agent and then the random agent with goal information. For tasks 4 and 5 only the bread crumb agent was able to complete the levels. Task 4 typically resulted in the random and ping agents getting stuck or falling in the water. Task 5 involved finding a staircase or ladder, which the random and ping agents rarely find.

We do not show the results with the path-planning agent, because they are typically longer times, because the path planner generates many more bread crumbs, each possibly requiring slight turns. For the handcrafted bread crumbs, only 2-4 are needed for all tasks. However, there were some scenarios in which the bread crumb agent became stuck; whereas, the path-planner agent was able to successfully complete the task. In these same cases where the bread crumb agent got stuck, we allow the attention agent to weight the random agents' actions more heavily, which was typically enough to get the player unstuck. Once unstuck, the attention agent readjusted the weight back in favor of the bread-crumb agent in order to complete the task. While this policy was hard-coded, it illustrates how multiple agent approach can combine to accomplish tasks not possible by a single agent approach and illustrates an area in which the DCA can focus efforts to learn such a policy.

6 Conclusions

We have successfully integrated the D'Artagnan Cognitive Architecture (DCA) to the Urban Terror (UrT) first-person shooter game to support the further development of the DCA. The definition of increasingly difficult tasks within the realistic maps provided with UrT comprises a challenging evaluation testbed for the DCA and other AI methods. The implementation and combination of the path planner and reflex agents provides a simple, yet effective, agent for playing our simplified UrT game.

We plan to pursue this work along three directions. First, we will further develop the various components of the DCA based on the latest understanding from cognitive and neurosciences. We will also interface and evaluate the DCA in other environments. Our goal is not only to produce a DCA-based agent that performs well in complicated environments, but that also exhibits human-consistent behavior. Second, we will extend the testbed to include additional tasks of increasing complexity by combining components of existing tasks and introducing adversaries. Third, we will make the UrT interface and task set available to others as a testbed for evaluating AI methods and as a mechanism to collect human play to serve as a baseline for measuring human consistency.

References

[Hill, 2002] R. Hill, C. Han and M. van Lent. Applying perceptually driven cognitive mapping to virtual urban environments. *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, pp. 886-893, 2002.

[Kondeti, 2005] B. Kondeti. Integration of the D'Artagnan Cognitive Architecture with Real-Time Simulated Environments. M.S. thesis, Department of Computer Science and Engineering, University of Texas at Arlington, 2005.

[Laird, 2002] J. Laird. Research in Human-Level AI Using Computer Games. *Communications of the ACM,* 45(1): 32-35, 2002.

[Laird and van Lent, 2001] J. Laird and M. van Lent. Human-level AI's Killer Application: Interactive Computer Games. *AI Magazine*, 22:15-25, 2001.

[O'Neill, 2004] J. O'Neill. Efficient Navigation Mesh Implementation. *Journal of Game Development*, Volume 1, Issue 1, 2004.

[Nallacharu, 2005] M. Nallacharu. Spatial Reasoning for Real-Time Simulated Environments. M.S. thesis, Department of Computer Science and Engineering, University of Texas at Arlington, 2005.

[Reynolds, 1999] C. Reynolds. Steering Behaviors for Autonomous Characters. Proceedings of the Game Developers Conference, pp. 763-782, 1999.

[Youngblood, 2002]. G. M. Youngblood. Agent-Based Simulated Cognitive Intelligence in a Real-Time First-Person Entertainment-Based Artificial Environment. M.S. thesis, Department of Computer Science and Engineering, University of Texas at Arlington, 2002.

[Youngblood and Holder, 2003] G. M. Youngblood and L. B. Holder. Evaluating Human-Consistent Behavior in a Real-time First-person Entertainment-based Artificial Environment. *Proceedings of the Sixteenth International FLAIRS Conference*, pp. 32-36, 2003.