# An Intelligent Action Algorithm for Virtual Human Agents

*Cagatay Undeger*
*Veysi Isler*
Department of Computer Engineering
Middle East Technical University
06531 Ankara TURKEY
+90-532-664-9015, +90-312-210-5579
undeger@metu.edu.tr, isler@ceng.metu.edu.tr

*Maj. Ziya Ipekkan*
General Plans and Policy Division
Technology, Defense Research and Armament Department
06100 Bakanliklar Ankara TURKEY
+90-312-402-1387
zipekkan@tsk.mil.tr

Keywords:
Computer Generated Forces, Virtual Human Agents, Mission Planning

**ABSTRACT**: *The objective of this study is to develop an intelligent action algorithm for virtual human agents on three-dimensional large terrains to accomplish a specified mission by group communication and coordination. The area contains natural and build-in entities such as trees, rocks, rivers, roads, houses, bridges, etc. Our platoons that are represented by virtual human agents enter a specific area to perform a specified mission, which may be to attack, escape or just pass through a selected tactical area. The area contains static and/or moving platforms such as jeeps, planes, helicopters, commandos, and etc. The goal of the agents is to complete their mission in a group or by being divided into groups of two or more without being detected or caught by a platform that carries different kinds of sensors (Day TV, Infra-Red, SAR). The output views of the platform sensors are observed by the user at tactical command center in order to make the detection process realistic. Agents may follow rivers, go through the forests, and hide behind trees, run, or even crawl in order not to be seen. When any of the agents are detected and identified, they try to escape or hide to complete their mission until they are caught or terminated.*

## 1. Introduction

Multi agent simulations are used to test real world behaviors and situations in computer-generated environments where intelligent agents react suitably to various events. Multi agent simulations are divided into two categories: on-line and off-line. Off-line simulations often use pre-processing and learning by fault techniques. To decide on the best strategy for a static situation, a set of generated plans is evaluated to choose the most suitable one for the current situation. Such off-line simulations can be ideal for tactical planning. In on-line simulations, a dynamic environment is tested in real-time. The agents learn about the environment by time and react to the events in real-time by using their knowledgebase. Such on-line simulations can be ideal for generating land, sea, and air battlefields, or testing sensor systems.

Agents are expected to learn about the environment, terrain and moving entities, and react according to changing situations under some assumptions. This is sometimes called “Reactive Planning”. In fact, the problem is how to gather environment information under which assumptions, how to store them, how to evaluate them for decision-making, and how to react suitably based on some goals. In this paper, a set of solutions is proposed for the mentioned problems.

The paper is organized as follows: in Section 2, a survey on related work is given, In Section 3, the

proposed approach is described in detail. The implementation of the prototype and a sample scenario are discussed in Section 4. Finally, Section 5 contains the conclusion.

## 2. Related Work

Intelligent systems are used in various domains such as robotics, computer generated forces, RoboCup soccer simulators. In robotics, intelligent planning aims to find out ways for interacting with the physical world which makes the problem hard to solve. In contrast, intelligent planning for computer generated forces aims to generate behaviors similar to the real world in a virtual environment. Simulating real world actions in a virtual environment is basically used to test some conditions that are not possible in the real world. Intelligent agents that behave much like humans are frequently used for pilot trainings in flight simulators[1,2]. In such simulations, realistic modeling of agent behaviors is important for the success of training. In order to be able to plan actions realistically, deciding on appropriate parameters, modeling environment, and using suitable algorithms for gathering information are very important. In 1999, the Defence Modelling and Simulation Office (DMSO) of USA established a working group of government and industry representatives and tied to decide the standarts of intelligent agents within the High Level Architecture, HLA [3].

In such applications, 3D environments are usually represented using polygons or DTED matrix. In the proposed approach of Champhell[4], terrain database and features are stored using triangles. In order to efficiently access information about roads, forests, buildings, rivers etc., spatial organization is utilized for representing the terrain. In our proposed technique, DTED matrix is used for calculating line of sight, planning path and generating slope matrix for the sake of efficiency.

In order to gather necessary information from the virtual environment, physical or stochastic methods can be used. In the proposed technique of Knuffner[5], a physically based method is used. To collect information from the 3D environment and to check which objects are visible to a particular character, the scene is rendered off-screen from the character's point of view, using flat shading with an unique color (object ID) for each object. Knowledgebase of agents are organized as link lists to store the information about the objects that are seen. Our proposed approach makes use of stochastic perceptions for gathering environment information. Visual perception is simulated using some criteria such as being in line of sight and viewing angle, range, volume, moving state, and plant cover density. Similarly audio perception is also simulated stochastically using information such as range and being in line of sight.

In multi agent simulations, evaluating the environment information and learning in time is essential. Erol Gelenbe proposed modelling computer generated forces with learning stochastic finite-state machines whose state transitions controled by state and signal dependent random neural networks[6]. In Knuffner's approach[5], rendering off-screen from the character's point of view and real-time path planning is used. His path-planning module aims to find a collision free path between a starting and ending point over the 3D terrain. In the study, the terrain is divided into embedded graph cells, which have vertical, horizontal (cost=1) and diagonal (cost=1.4) costs of walking through. The suitable path is found using Dijkstra's algorithm by minimizing the total cost. Path planning is categorized into two: real-time and off-line planning. The above method is considered to be real-time path planning. In off-line systems, it is enough to find a path in advance for a static environment, but in real-time systems, it is essential to use an efficient algorithm that considers the environmental changes in order to find a collision free and threat safe path [7,8].

In addition to path planning, many agent simulation systems have a module called "Reactive Planning" for the purpose of deciding and reacting efficiently to various events using a rule set considering the goals, knowledgebase and previous experiences[9,10]. We decided to use reactive planning module in our simulation environment using a goal-directed decision tree in order to react efficiently to new detected perceptions such as sensor platforms.

Group coordination is also an important concept in multi agent simulations. Without coordination, the agents can only be considered as individual groups with no relation. Baxter and Horn[11], organized a command hierarchy used by the agents which based upon the military command structure. In the hierarchy, the groups are under control of a squadron commander. In addition, groups have their own troop commanders. Organizing the groups along the same lines as the military formations allows emulating the change of getting plausible behaviors. It also provides a framework to guide the comminication between the agents and allows the planning of complex group

orders to be divided into several smaller problems. Communication of orders passes straight down the hierarchy and intelligent information is shared between peers and comminicated to superiors. The commanders are responsible for gathering information about their own situation, passing it up to their superiors and peers, and giving orders to their subordinates to achieve the commander's high level objective. That structure shows similarity to our proposed group and command structure. We have organized the agents in groups with a commander to achieve a specific group mission. Commanders are responsible for gathering environment information, passing it up to other group commanders by radio messages and controlling the group. The subordinates are resposible for following their group commanders and executing the commander orders.

## 3. Proposed Approach

This study involves modeling and representing actions for virtual human agents that should accomplish a given tactical mission in a virtual battlefield, which is a part of a sensor simulation system. The objective of developed software is to test a sensor optimization algorithm using realistic scenarios that are executed by our intelligent human agents. The main goal of the agents is to accomplish the given mission without being detected or caught by a sensor platform.

### 3.1 Agent Groups

Agents are organized in groups that are moving under the control of a group commander. All the moving entities, intelligent agents, sensor platforms, and animals, are considered in groups and kept in the same data structure. Groups are classified into three, red, blue and white groups. Red groups are our intelligent agents. Blue groups are threats that are sensor platforms, and white groups are animals that are used to make the agent perceptions go wrong. The hierarchal structure of world, groups and environment is shown in Figure 3.3.1.

We assume that the terrain and plant cover information is known and the agents hear all the radio messages.

### 3.2 Terrain Structure

Terrain information is stored in a DTED matrix. Using this matrix, a slope matrix containing slope directions and magnitudes is calculated. Slope magnitude is a real number between 0 and 1 where 0 indicates that there is no slope and 1 indicates that there is a 90-degree slope. We assume that the agents moving along the slope direction slow down and consume more energy while the agents moving along the opposite slope direction go more easily and spends less energy. The slope has no effect on an agent moving perpendicular to the slope direction. Calculation of slope direction and magnitude is shown in Figure 3.2.1.

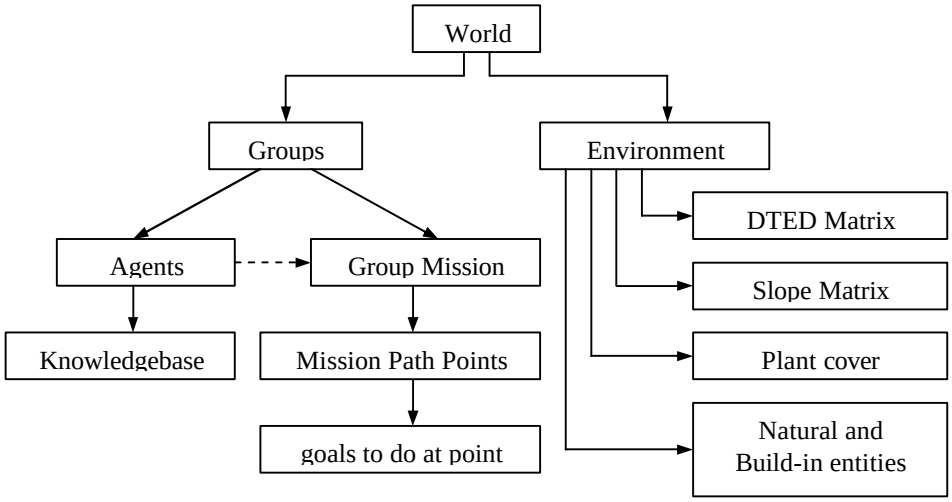


Figure 3.1.1. The hierarchical structure of world, groups and the environment

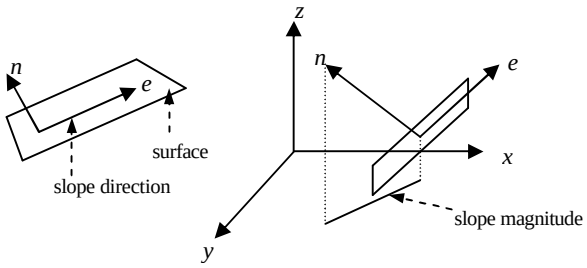


Figure 3.2.1. Calculation of slope direction and magnitude

Terrain plant cover is stored in polygons. Every polygon determines a limited area and defines terrain structure (plant density, average plant cover height, etc.) related to this area. Features like rivers and bridges are also stored in this way. These features also have 3D polygonal information.

### 3.3 Goal Description

In the proposed approach, the goal belongs to the group. Every group has its own goal plan (group mission) and moves with its commander's orders. The group commander gives decisions to accomplish the mission and subordinates follow their commander under normal conditions. Goal description is given as a set of path control points and a list of goals to be achieved at these points. Path control points are categorized into five group which are starting point, target point, home point, pass through points and tactical points. To reach these control points, commander generates a path considering the terrain and detected threat information. Unless an abnormal situation occurs, the commander follows that path.

Figure 3.3.1. A screen snapshot of a sample scenario

At tactical control points, a set of goals (send radio message, put bomb, etc.) can be given. Control point coordination is also handled using these goal lists. Goal items define the activities to do on specific events such as arriving at a control point, waiting at a control point, or leaving from a control point.

### 3.4 Stochastic Based Seeing and Hearing

Agents gather information from the environment by seeing and hearing based on probabilistic computations, but that doesn't mean that the sensor detections work stochastically. The sensor perceptions are calculated and sent to agents even the probability is very low. The agents test the probability. If the perception is owned by an unnoticed entity, a probability test is done before accepting it. If the agent decides to notice the entity, then the entity is always seen or heard without calculating the probability until it is away from the agent's point of view. That can be described by the following example. A person always looks around. Eyes capture everything that is possible to be seen, but human may not notice them because of his low attention. Once an entity is noticed by the help of attention, seeing and following it becomes continuous. Hearing can be considered similarly. The difference is that hearing doesn't depend on being in viewing angle. The important parameter is range and being in line of sight.

Probability of detection depends on the following parameters:

- Being at line of sight
- Being in viewing angle
- Volume
- Range
- Movement
- Plant cover
- Weather condition
- Noise

Being in line of sight means, the target is not occluded by any object. In order to test the line of sight, DTED matrix is used instead of 3D terrain information. But the use of DTED matrix is not enough. We have to perform another line of sight test for the natural and build-in features in the environment.

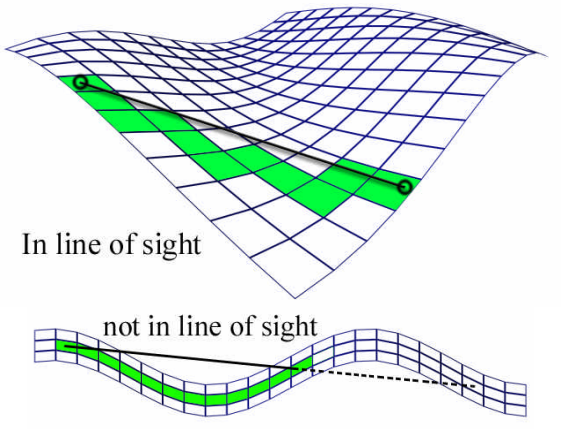


Figure 3.4.1 Calculating line of sight from the DTED matrix

There is no probability of detection for the objects, which are neither in the line of sight nor viewing angle. But the objects can be sensed using audio cues. If an object is both in line of sight and viewing angle, then terrain plant cover, the position of the object in the viewing frustum, range, volume, and movement of the object forms a probability of detection. Closeness to the line passing through the focus of the viewing frustum and to the agent, speed of the movement, low plant cover density and high volume are the criteria that increase the probability.

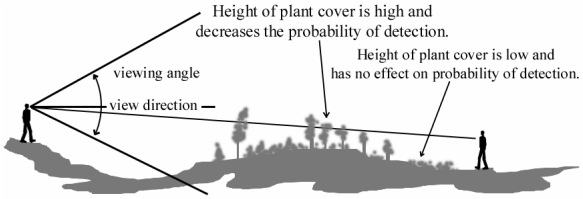


Figure 3.4.2. The effect of plant cover density and viewing angle

Hearing is modeled as a probability function depending on the range and the speed of the movement of the object that causes noise. The noise from a truck is because of its engine while the noise of a human is caused because of his steps. If the object is not in the line of sight e.g. the target is behind a wall, the probability of hearing is decreased.

In order to increase the frustum of agents, head movements are added to the agent model. For some specific conditions, agents move their heads in different ways to collect more information. In normal conditions the agents look around and also look at their commanders and if there is a new perception, they change their head directions to this new detected object.

### 3.5 Classifying and Storing Perceptions

The gathered information from seeing and hearing is classified into three categories by using range and volume. These are detection, recognition and identification. Classification criteria are shown in Figure 3.5.1.

If a new perception is detected, it is added to the knowledgebase of the agent. The knowledgebase is stored in a dynamic link list. The problem is to decide whether the perception is a new object or an update of a previous detected object. Although we are in a computer-generated world and have the information of all the objects in the environment, the ID of detected object is not sent to the agent unless it is identified. The agents have to find the similarities themselves and

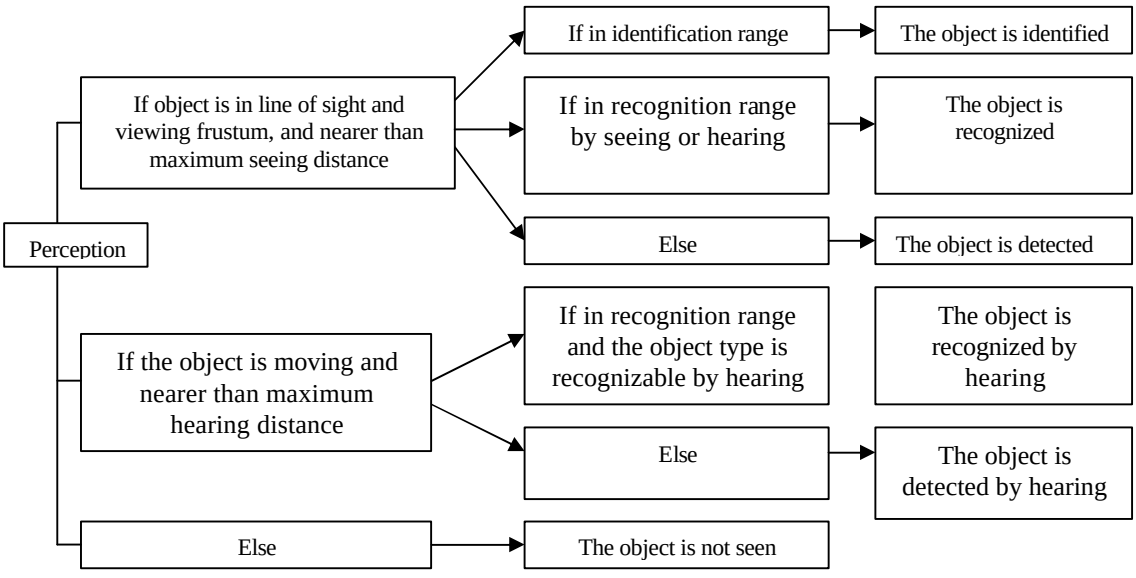




Figure 3.5.1. Perception is classified into three categories: detection, recognition and identification

update the knowledgebase. The similarity is found out using estimated positions of previous detected objects and the positions of new detected objects. If a similarity is searched for previous detected object information in the knowledgebase, an estimate of position is calculated using previous movement threshold that is calculated considering the previous speed, we may accept that they are similar. If similarity is found the previous knowledgebase is updated using new detected perception, orherwise the perception is added to the knowledgebase as a new item.

| Main loop |
|---|
| For each group<br>For each agent<br>If the agent is a red team member (intelligent agent)<br>Construct the sensor detection list (perceptions) for the agent; > (A)<br>Analyse the detected list and update the knowledgebase; > (B)<br>Execute behavior module; > (C)<br>Update the pysical appearance; |
| A |
| Backup the previous detected list and create a new list;<br>For each group<br>For each agent (target) except himself<br>Calculate the seeing and hearing statistics between the agent and the target; > (D) |
| B |
| For each member of detected list<br>If the detection exists in the previous list and unsensed because of the probability test and no probability change occurred after that time<br>Mark the member of new detected list as “unsensed”;<br>For each sensed member of new detected list<br>Do a comparison to knowledgebase, find the similarities;<br>If similarity found<br>Update the knowledge and check the member of detected list as “similarity_found”;<br>For each member of new detected list which is not checked as “similarity_found”<br>Do a probability test and if it is passed<br>Add the list member to the knowledgebase as a new perception;<br>Else<br>Check the member of detected list as “unsensed”; |
| C |
| Find who the commander is;<br>Find the status of the mission plan;<br>If the agent is a commander<br>If there is no abnormal condition<br>Follow the path;<br>Else<br>Execute the reaction planning module;<br>Else<br>If there is no abnormal condition<br>If the commander position is known<br>Follow the commander;<br>Else<br>Stop and search for the commander;<br>Else<br>Execute the reaction planning module; |
| D |
| Compute the statistics between the agent and the target (line of sight, viewing angle, range, etc.);<br>If there is any possibility of detection, add the perception to the detected list; |

Table 3.6.1 Basic lines of the decision and reaction algorithm

direction. If the range between the estimated position and the new detected object position is smaller than a

### 3.6 Decision and Reaction

Agents give decisions and react to various events using only their knowledgebase. In addition to object positions, knowledgebase also contains the information of who the commander is and the radio and face-to-face messages. By using this information, agents find out their commander and the status of the mission plan. If an agent is a commander, it executes the mission otherwise it follows the commander. If an agent doesn't know the position of his commander, it stops and tries to find out the commander by radio messages.

Route between control points is generated using "Path Planning Module". Unless an abnormal situation occurs, the commander follows that path. Otherwise, until the conditions become normal, "the Reaction Planning Module" is executed by canceling the path. The main objective of reaction planning module is not to be seen by any sensor platform. To follow that objective, the agent may decide to slow down, stop, run, crawl, etc. For executing that purpose a decision tree and a short distance path-planning module is used. Having returned to normal conditions, a new path planning is generated using "Path Planning Module" considering the new world information.

### 3.7 Group Coordination

Every group has a mission, which is a part of a high level mission. As mentioned in Section 3.3, the mission is given using a set of control points and a list of goals to be achieved at that point. Some of these goal items contain radio messages to be sent at the points. Radio messages are used for coordinating the groups or giving information to other groups about detected threats. Seven basic events may happen at a point. These are arriving, waiting, becoming ready to leave, waiting for a while before leaving, leaving, canceling mission, being waited too much. The goal list of a control point describes the actions to be done if any of these events occurs.

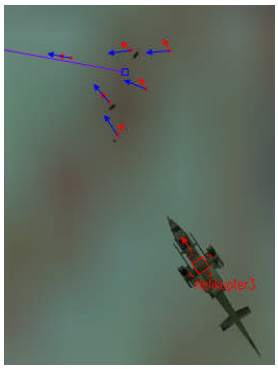

Figure 3. A group having six agents is walking in a formation. Walking and looking directions of agents are shown with different arrows.

In addition to coordination between groups, there is also coordination and a group formation for positioning inside the group. The members of the groups may talk to each other and notify them about the threats that are detected. If it is needed, the commander passes this information to other groups. A group formation is used in order to make the members move in-group. The position of a member in the group is defined by relative (x,y) coordinates to the commander. The member coordinates are calculated using the current position and direction of the commander and this calculated position is given to the member as a target position to be reached. If a subordinate is far from its expected position, it runs for a while to take the right position.

### 3.8 Physical Modeling

A 3D model is generated for the agents in order to be sensed by the platform sensors. For the physical status of an agent, the coordinates, the posture, the status of posture transition and the head direction is used. In addition, a collision detection algorithm is needed for the physically modeled agents. In our model, a cylinder bounding volume is used for that purpose.

## 4. Implementation

The implementation is done on SGI ONYXII and O2 platforms using C++ and Vega Paradigm (based on IRIS Performer). The 3D terrain is created from the DTED information and saved as Open Flight File format. Sample scenarios are generated in order to test the agent actions. In the paper, we have proposed an algorithm that searches for similarities in the knowledgebase. The test of this algorithm shows that in some extreme conditions, the method may generate

wrong perception links to knowledgebase. But in general, it works fine. A brief sample scenario input is given and described in Table 4.1.

## 5. Conclusion

This paper presents a multi agent model for testing a sensor simulation system. The agents are organized in groups whose goal is to accomplish a specified mission by group communication and coordination. The high level mission is divided into smaller plans and assigned to each group. Group plans are defined by a set of control points that have a list of goals to do at. The agents gather information about the environment using sensor systems that work stochastically. The gathered information is evaluated using an algorithm that searches for similarities in knowledgebase. The behaviors are generated using path planning, reactive planning and decision trees.

## 6. References


[1] Randolph M. Jones, John E. Laird, Milind Tambe, and Paul S. Rosenbloom: "Generating Behavior in Response to Interacting Goals" Proceedings of 4th conference on Computer Gererated Forces and Behavioral Representation. Orlando, Florida, 1994.

[2] Randolph M. Jones, Milind Tambe, John E. Laird, and Paul S. Rosenbloom: "Intelligent Automated Agents for Flight Training Simulator" Proceedings of 3th conference on Computer Gererated Forces and Behavioral Representation. Orlando, Florida, pp. 33-42, May 1993.


| Description | Peusedo Code | |
|---|---|---|
| Create the world database. Create the DTED matrix | cPlatoon * plt; cAgent * agent; cGoalPoint * point; cGoalItem * item<br>world->Initialize();<br>world->Create_World(...);<br>world->Create_DTED(...); | |
| Add a group and two agents to the environment | plt = world->Add_Group( GroupRed );<br>agent = plt->Add_Agent(...);<br>agent = plt->Add_Agent(...); | |
| Set group position and formation of the agents | plt->SetGroupPosition( 9163.0, 10259.0 );<br>plt->Set_Agents_Group_Formation(8,true); | |
| Get the starting control point. Add a goal item for the point. Move fast to the next point | point = plt->plt_goals->GetObjectFromIndex(0);<br>point->movingstate = MovingFast; | |
| | Goal item 1.1 | Item=point->Add_GoalItem(giWaitUntilReceiveKeyword);<br>item->data.ReceiveFromID = 1;<br>item->SetKeyword("First Step Go..."); |
| Add a control point | point = plt->Add_Plt_GoalPoint( gpPassThrough, 9345.0, 9742.0 ); | |
| Add a tactical control point. Wait until the group 2 leaves from point 3 | point = plt->Add_Plt_GoalPoint( gpTactical , 9220.0, 9595.0 ); | |
| | Goal item 3.1 | item= point ->Add_GoalItem(giWaitUntilContinueMission);<br>item->data.ReceiveFromID = 2;<br>item->data.GoalPointID = 3; |
| Add another tactical control point. Wait until group 1 arrives at point 5. When ready to continue, send a keyword to group 1. Wait for a response from group 1 to leave the point. Move slow to the next point | point = plt->Add_Plt_GoalPoint( gpTactical , 8945.0, 9270.0 );<br>point ->movingstate = MovingSlow; | |
| | Goal item 4.1 | item = point ->Add_GoalItem(giWaitUntilArrival);<br>item->data.ReceiveFromID = 1;<br>item->data.GoalPointID = 5; |
| | Goal item 4.2 | item = point ->Add_GoalItem(giReadyToContinueDo);<br>item->data.doAction = doSendKeyword;<br>item->data.SendToID = 1;<br>item->SetKeyword("Group 3 Ready"); |
| | Goal item 4.3 | item = point ->Add_GoalItem(giWaitUntilResponseKeyword);<br>item->data.ReceiveFromID = 1;<br>item->SetKeyword("Mission Start"); |
| Add the target point. Put a bomb to the point and move fast after leaving. While leaving the point send the keyword "mission completed" | point = plt->Add_Plt_GoalPoint( gpTarget , 8935.0, 9130.0 );<br>point ->movingstate = MovingFast; | |
| | Goal item 5.1 | item = point ->Add_GoalItem(giArrivalDo);<br>item->data.doAction = doPutBomb; |
| | Goal item 5.2 | item = point ->Add_GoalItem(giContinueMissionDo);<br>item->data.doAction = doSendKeyword;<br>item->data.SendToID = -1;<br>item->SetKeyword("Mission Complete"); |
| Add another control point | point = plt->Add_Plt_GoalPoint( gpPassThrough, 9212.0, 9124.0 ); | |
| Home point | point = plt->Add_Plt_GoalPoint( gpHome , 9683.0, 8400.0 ); | |

Table 4.1. A segment of sample scenario input

[3] Paul T. Barham and Shirley M. Pratt: "The Development of High Level Architecture (HLA) Human Starter Simulation Object Model (SOM)" Proceedings of 8th conference on Computer Gererated Forces and Behavioral Representation. Orlando, Florida, pp. 145-151, May 1999.

[4] Charles E. Campbell and Michael A. Craft: "Advancements in Synthetic Natual Environment Representation" Proceedings of 8th conference on Computer Gererated Forces and Behavioral Representation. Orlando, Florida, pp. 81-86, May 1999.

[5] James J. Kuffner, jr. and Jean-Claude Latombe: "Fast Synthetic Vision, Memory, and Learning Models for Virtual Humans" Proc. of Computer Animation, IEEE, pp. 118-127, May 1999.

[6] Erol Gelenbe: "Modelling CGF with Learning Stochastic Finite-State Machines" Proceedings of 8th conference on Computer Gererated Forces and Behavioral Representation. Orlando, Florida, pp. 113-115, May 1999.

[7] James J. Kuffner, Jr. and Jean-Claude Latombe: "Goal-Directed Navigation for Animated Characters Using Real-Time Path Planning and Control" Proc. of CAPTECH '98: Workshop on Modelling and Motion Capture Techniques for Virtual Environments, Geneva, Switzerland, pp. 26-28, Nov 1998.

[8] Kazuo Sugihara and John K. Smith: "Genetic Algorithms for Adaptive Planning of Path and Trajectory of a Mobile Robot in 2D Terrains" Technical Report, number ICS-TR-97-04, University of Hawaii, Department of Information and Computer Sciences, May 1997.

[9] Jin Joe Lee and Paul A. Fishwick: "Real-Time Simulation-Based Planning for Computer Generated Force Simulation", Simulation, pp. 299-315, 1994.

[10] Rune M. Jensen and Manuela M. Veloso: "Interleaving Deliberative and Reactive Planning in Dynamic Multi-Agent Domains" AAAI Fall Symposium: Integrated Planning for Autonomous Agent Architectures, October 1998.

[11] Jeremy W. Baxter and Graham S. Horn: "A Model for Co-ordination and Co-operation Between CGF Agents" Proceedings of 8th conference on Computer Gererated Forces and Behavioral Representation. Orlando, Florida, pp. 101-111, May 1999.

## Author Biographies

**CAGATAY UNDEGER** is research assistant in Department of Computer Engineering, Middle East Technical University. He is working in the Modeling and Simulation Laboratory as part of his master thesis.

**VEYSI ISLER** is a faculty member of the Department of Computer Engineering, Middle East Technical University (METU). He received his B.Sc. degree in Computer Engineering from the same university, in 1987. He worked as a research assistant and instructor for the Department of Computer Engineering and Information Sciences, at Bilkent University where he received his M.S. and Ph.D. degrees between 1987 and 1995. He is the Coordinator of Virtual Environments Group in Modeling and Simulation Laboratory of METU.

**ZIYA IPEKKAN** received M.S. degree in OR from Naval Postgraduate School, Monterey, USA, in 1989. He initiated development of several models, approaches and solutions to assessment and evaluation of force structures. He is currently responsible for Modeling and Simulation activities within Turkish Armed Forces.