Military Technical College
Kobry El-Kobba
Cairo, Egypt

12-th International Conference
on
Aerospace Sciences &
Aviation Technology

A DISTRIBUTED INTERACTIVE SIMULATION FOR A TANK-SHOOTING TRAINER

A.F.SELEEM*, H.A.ALY**,E.TALKHAN***.

ABSTRACT

This paper presents design and implementation of a Distributed Interactive Simulation (DIS) for a tank-shooting trainer. The developed system involves a highly modular collection of software packages designed to facilitate the creation of devices, it also involves a hardware interface between hardware fabrications - a circuit of microcontroller - and the used integrated software packages. In our system we simulate the gunner's station only in a virtual environment. This paper gives an analysis of the main features of the real system, the design of our proposed simulator parts and it's operation, and the software implementation that is used for integrating all the functions of our proposed simulator.

KEY WORDS

Distributed Interactive Simulation(DIS), Virtual Environments, Gunner's Station, Tank Simulator , Gunner's Power Control Handles, Gunner's Primary Sight, Simulator Design, VEGA, LYNX, Application Definition Files (ADFs).

---

* Graduate student, Egyptian Armed Forces, Cairo, Egypt.
** PhD, Dept. of Computer, Military Technical College, Cairo, Egypt.
***Associate professor, Dept. of Computer Engineering, Cairo university, Cairo, Egypt

1. INTRODUCTION

Distributed Interactive Simulation (DIS) is an infrastructure for building large-scale virtual worlds by interconnecting independent simulators via a network. DIS has been used for training military personnel, where the virtual world is a battlefield. Models and simulations are used for several important purposes: training to maintain readiness, analysis of the effects of proposed tactics or system acquisitions, planning and rehearsal of operations, and demonstration of new technologies[1]. We will use DIS to simulate the training of the two actual tank gunners to be trained on gunner's station simulator via a network as shown in Fig.1. Old systems as VEGS simulators have one trainer only and it also work with legacy techniques on magnetic tapes with few number of missions and with limited number of environments .
Our distributed interactive tank simulator has some objectives which are :

i- Uses our environmental effects without constraints.

ii- Realistic interactive training with native language(Arabic) user interface .

iii- Training on local-typed terrains in a variety of weather conditions.

2. ANALYSIS

In our analysis of the proposed simulator, we followed the steps mentioned in[3] which are :

i- Formulate the problem and plane the study.
ii- Collect data and define a model.
iii- Construct a computer program and verify.
iv- Make Pilot Runs.
v- Validate?
vi- Make production runs.
vii- Analyze output data.
viii- Document, present, and implement results.

In our design the main features of the combat tank considered consists of the hull and turret assemblies. The turret can rotate a full 360 degrees. All system connections between hull and turret are made through a slipring mechanism. The tank uses high speed, maneuverability, and a variety of weapons to attack and destroy enemy tanks, equipments, and forces. The tank provides protection from enemy weapons. The main design of the tank have four components as illustrated in Fig.2. : driver's station, commanders station, gunner's station, and loader's station. "Weapons exist in two main combat rounds, generally known as sabot and heat rounds (for high-explosive anti-tank), inflict damage in very different ways"[7].

3. TANK SIMULATOR DESIGN

In our simulator, we only modeled the gunner's station with all the necessary software and hardware ,we assumed three main parts, as depicted in Fig.2 :-

1- Gunner's Station.
   a- Gunner's Power Control Handles.
   b- Gunner's Primary Sight.
2- Hardware Interface.
   a- Interface between Power Control Handles and Software program.
   b- Interface between Gunner's Primary Sight and Software program .
3- Software Program.

For the gunner's Power Control Handles we will describe the power control handles as in [7] and as in Fig.3 ; it consists of :

- grasp power control handles (5) and squeeze either or both palm switches (4) (either palm switch on gunner's power control handles must be on for any function of handles to operate) laser buttons (6), and triggers (7) . It operates as follows :

- Turn the handles clockwise to traverse turret right and counter- clockwise to traverse turret left.

- Rotate handles back to elevate weapons, and rotate it forward to depress weapons.

- Operate the laser range finder by pressing either or both laser buttons on top of power control handles. The range switch on the Laser Range Finder (LRF) must be set to ARM 1ST RTN or ARM LAST RTN mode to get the accurate range.

As for the gunner's primary sight, it's normally the sighting Instrument for main and coaxial guns[7]. The eyepiece for the gunner's primary sight periscope sight is at the gunner's station. The gunner's primary sight includes LRF transceiver and a thermal night vision subsystem. The gunner's primary sight offers a range of daylight vision choices magnification powers of 3X and 10X which can be selected when sighting through eyepieces. its symbology is shown in Fig.4. Gunner's Primary Sight consists of

the upper and lower panels and a Thermal Image System (TIS). Each panel has its own switches, knobs, lights perform functions as mentioned in Table 1.[7] .

Both components (The gunner's primary sight and gunner's power control handles) send their status and values to the software programs through an interface circuit. We designed this circuit using a chip microchip PIC 16F877 to read the status of :

- 5 switches , each switch has three states.
- 5 switches , each switch has two states.
- 1 rotary switch with five states.
- 1 bush button.
- 6 knobs.
- 10 lamps.
- Control Handles direction

The read data is sent to the host PC via the PS/2 port. The micro controller chip is connected using the serial peripheral interface(SPI)protocol which is implemented on the chip.

These reading of the hardware devices are written to a text file or a database which are then used by software program to get all the values and the status of hardware controls.

4. SOFTWARE IMPLEMENTATION

The software components used in our proposed distributed real-time interactive simulation are : the modeling package(MultiGen Creator), software visualization environment (VEGA),and our developed real-time application program(using C language).

4.1 The Modeling Package

The Create Object function is used to create different objects in the visual scene as required if the 3-D model of the required object was built by a third-party packages. The 3-D model is built in our simulation either by 3-DMAX or MultiGen Creator but the later is better because it makes very light models which indicate better performance in visualization. The CreateObject is also used to set the initial position of the created object in all three axes and its orientations. The X, Y, and Z coordinates values define an offset along these axes. In MultiGen Creator, a right-hand coordinate system as in Fig.5. is used, i.e, positive Z is up, positive Y is forward, positive X is right. the H, P, and R values represents the orientations of the heading, pitch, and roll of the object respectively. Positive heading, skews the object to the left, Positive pitch, skews the object up, and Positive roll, skews the object counterclockwise[5].

In our work ,We use MultiGen Creator(Its file format is *.flt) to model the following objects :

1- Land
2- Friend tank
3- Enemy tank
4- Tank after distortion
5- Gunner's primary sight symbology

4.2 Software Environment (VEGA)

VEGA™ is MultiGen-Paradigm's premier software environment for the creation and deployment of real-time visual and audio simulation, sensor, virtual reality, and general visualization applications. By combining advanced simulation functionality, VEGA provides the basis for the most productive process for building, editing, running and deploying sophisticated applications quickly and easily[6]. VEGA is intended to be used in conjunction with LynX, a graphical user interface for defining and previewing VEGA applications. Although VEGA contains all of the application programming interface API necessary to create an application, LynX simplifies the development process. LynX allows the user to configure an application with a tree view as given in Fig.6 without writing code. In many cases, it is helpful to use both LynX and VEGA API in an application. VEGA functionality can be extended by additional special purpose modules as in Fig.7. We can create our Application Definition File (ADF) as in Appendix.A.

4.3 The Realtime Application Program

This program controls the graphical scene, how the user moves through the scene, as well as a wide variety of other dynamic events during the scene. Tank driving , collision detection, and special effects such as explosions are also included in this real-time application program which generally consists of the C or C++ source code files (.dsp files), along with a workspace file (.dsw) as in Fig.8. When the header file(vg.h, in the case of VEGA) and library inclusion directories are set up properly in Visual C++ , the appropriate VEGA module libraries are automatically linked to our application .

The main functions that will be used in our applications, are described below

1- CreateObject : The CreateObject is used to create different objects in the visual scene as required if the 3-D model of the required object was built such as tank, vehicles, plane, trees, buildings, etc.
2- Commonpos_change : The Commonpos_change is used to change the position of any object in the visual scene in all three axes as required.
3- CreatePath : The CreatePath can create waypoints for a predetermined path and then control motion automatically by traversing the path with objects using a navigator.
4- Showlenght : The Showlenght is used when laser beam technology is

used in shooting to show the distance between the object that trigger laser and the object which is triggered at.

5- SensorVision : The SensorVision is used to allow the output of Observer to be rendered at any wavelength from the visible through the far infrared band.
6- Fog : The Fog is an environment effect to the atmospheric model. It is important to understand that the ground fog range and other parameters can be controlled by this function.
7- Storm : The Storm is used to model the environment effect of storms that requires values for parameters specifying size, visibility, and transition.
8- Ephemeris : The Ephemeris is used to affect the ephemeris environment effect which is capable of calculating realistic Sun and Moon positions, given the the time of day and date.
9- Deactivate : Deactivate It disable the motion of the user if damaged.
10- Reactivate : The Reactivate is used to reactivate the motion of the user.

Our tank simulator application program workflow

1- Include all header files needed for our application such as <stdio.h> , <stdlib.h >, <string.h> , <vg.h> , <vgfx.h> .
2- Include all functions needed for calling ADF file such as <vgaudio.h> , <vgnetdis.h> , <vgaudio.h>
3- Define all constant parameters and all variables needed for applications.
4- Load ADF file with vgDefineSys("c:\\tank.adf");.
5- Get the movement of Gunner's Power Control Handles from the external hardware database file .
6- Also, get all functions of gunner's primary sight such as (toggle 3x display - toggle 7x display , default environment - sensor environment , main gun – safe) from the external interface hardware database file .
7- Get the distance between objects (by using laser beam).
8- Define all the equations of all types of guns.
9- Control all environment effects (snow, rain, time of day…….etc).
10- All steps are repeated in a super loop(while (1)) to get a real-time application.

5. EXPREIMENTS AND RESULTS

Our experimental setup is as follows :

1- Connect the two trainer computers via the network.
2- Run the application program file on both computers and test for connection between the two computers via the network by moving one of the trainers on one side and follow the related effects on the other side.

3- Each trainer moves the control handles to control motion of primary sight in front of the scene as in Fig.9 to be able to acquire the enemy tank and fire.
4- Also, the trainer controls all parameters in the scene .
5- If the enemy tank is destroyed then the simulator program reports all actions of the two trainers and computes the scores of the trainer.

In the gunner exercises there are several levels of target acquisition. Each level increases in difficulty by reducing visibility and adding distractions for the crew. Each gunnery training exercise contains scored situations involving similar tactical conditions. A situation can contain one or more targets which are either stationary or moving. When the exercise is completed, the scores for each skill area are added together and averaged. The average score in each skill area is used to determine the level for the crew. Each situation is scored for performance by the crew in target acquisition and reticle aim. The score for each skill area is based on the items listed in the scoring criteria table below :

Skill Dimension	Criteria
Target Acquisition	Time to acquire target. Number of Identification/classification errors.
Reticle Aim	Time of fire first round/burst. Time to kill. Magnitude of aiming error (Main Gun only).
System Management	Number of switch setting errors prior to firing. Number of switch setting errors at the time of firing.

Target Acquisition (TA): Involves those skills required for the crew to accurately detect, identify, and classify targets. Target Acquisition time begins when the target is fully exposed and ends when the following command is logged . Target acquisition scoring is based on the following:

Acquisition Time (Seconds)	Evaluation
0-5	4
> 5-10	3
> 10-15	2
> 15	1

Reticle Aim (RA): Involves those skills required to lay the reticle on the proper target aiming point, fire at, and destroy a given target or targets. The crews Reticle Aim score will be based on the lowest score as follow :

Main Gun/COAX Time (Seconds)	Evaluation
0-13	4
> 13-18	3
> 18-23	2
> 23	1

System Management (SM): Evaluates the crew's ability to operate as a crew, utilizing the correct principles and techniques of gunnery. The score in system management is based on the lowest score as follow :

Number of Prefiring Switch Errors	Evaluation
> 13-18	3
> 18-23	2
> 23	1

After several runs for comparative studies on users skills enhancement using the proposed system compared to a legacy one we found that users were able to :

1- Enhance the recognition of all types of vehicles and tanks used in gunnery training exercises because :
   a- has more hundreds of vehicles or tanks(moving and stationary) where the old simulator is poor of these objects.
   b- defining any paths of objects where the old one has no path defining because all the paths is ready made.
   c- has realistic special effects such as : vehicle distortion, debris, smoke, fire, flame….etc, which all does not exist in the old system.
   d- has realistic lights such as moon light, sun light, spot light which does not exist in the old system.
2- train in many levels of exercises which start with easy levels and end with difficult levels.
3- train to follow the enemy tanks and fire in the required time.
4- develop their skills in using laser range finder to get the distance between the object that trigger laser and the object which is triggered at to fire before the triggered object change its place.
5- has the ability to change the training land of any exercises (agricultural or Desert) by selecting it before start training where the old simulator has ready made land with no change,
6- use our native lands with its details where the old simulator uses foreign lands.

Finally, our simulator has some advantages over legacy simulators such as :

Our simulator	Old simulators(such as VEGS)
More easy to use	Visibility problems
Simulate two interactive gunners	Simulate only one gunner
Use distributed interactive techniques Reduces bandwidth traffic consumption	No use of network traffic
Realistic visualization	Peer objects
Uses environmental effects	No use of Environmental effects
Upgradable by us	Use only external upgrades
Easy maintainable	Difficult to maintain
Uses Our native language	Uses foreign languages

6. CONCOLUSION AND FUTURE WORK

In this paper, the development of an interactive shooting simulator using real-time simulation and virtual reality applications is proposed. The developed system involves a highly modular collection of software packages designed to facilitate the creation of device independent virtual environments. The application programs can run on different hardware platforms ranging form both end-user programs and C++ APIs (Application Programming Interfaces). We involved hardware devices that meets the requirements of the tank gunner's primary sight panel using a hardware interface based on a new fabricated circuit of microcontroller. The performance of the developed system is investigated via the simulation of real gunner station of the tank and the obtained results affirmed the potential benefit of the developed system for interactive training application. The tank simulator could be used for training gunners on different exercises of realistic interactive training on all types of terrain in a variety of weather conditions. In future developments we will use the new technology of HomePlug Ethernet Adaptor which brings high speed networking using AC power wiring[9] .

7. REFRENCES

[1] Fujimoto, R. M., "Parallel Discrete Event Simulation", Communications of the ACM, Vol. 33, No. 10, Oct. 1990, pp. 30-53.

[2] Pamela McCauley Bell, Ph.D. , Rhonda Freeman , "Qualitative and Quantitative Indices for Simulation Systems in Distributed Interactive Simulation", 3rd International Symposium on uncertainty modling and analysis(ISUMA 95), 1995 IEEE. pp. 745-748.

[3] Averill M. Law ,and W. David Kelton,"Simulation Modeling & analysis book", McGraw-Hill International Editions, second edition,1991,pp. 106-109.

[4] R. Rajkumar, M. Gagliardi, and L. Sha, "The Real-Time Publisher/Subscriber Inter-Process Communication Model for Distributed Real-Time Systems: Design and Implementation," in First IEEE Real-Time Technology and Applications Symposium, May 1995.

[5]http://ww.paradigmsim.com/products/database/creator/index.shtml, the home page of MultiGen-paradigm's Creator product.

[6]http://ww.paradigmsim.com/products/runtime/vega/index.shtml, the home page of MultiGen- paradigm's Vega product.

[7] http://www.people.howstuffworks.com/m1-tank.html , documents the tank description.

[8] VanHook, D. J., Calvin, J. O., and Fusco,"An Approach to DIS Scalability", 11th Workshop on Standard for Inter-operability of Distributed Simulation, Vol. 2, 1994, pp. 347-356.

[9] http://www.smarthome.com/6405e, HomePlug Ethernet Adaptor User's Manual and Utility.

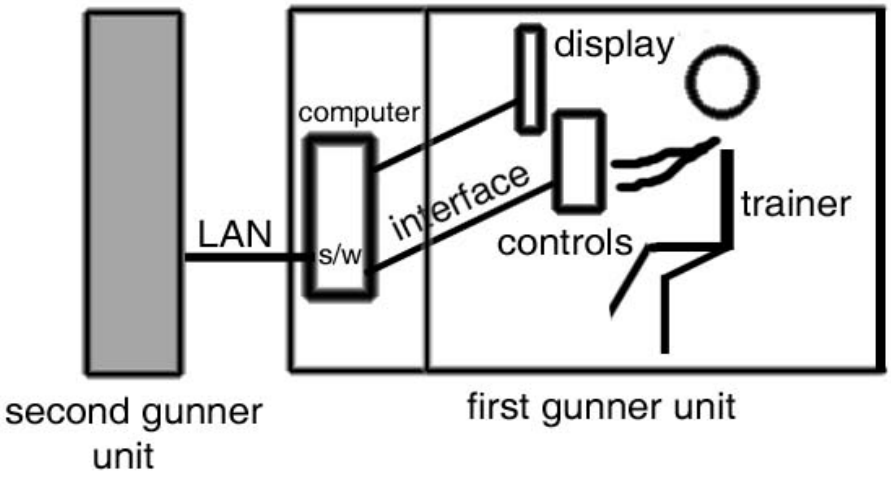

Fig.1. Tank simulator design

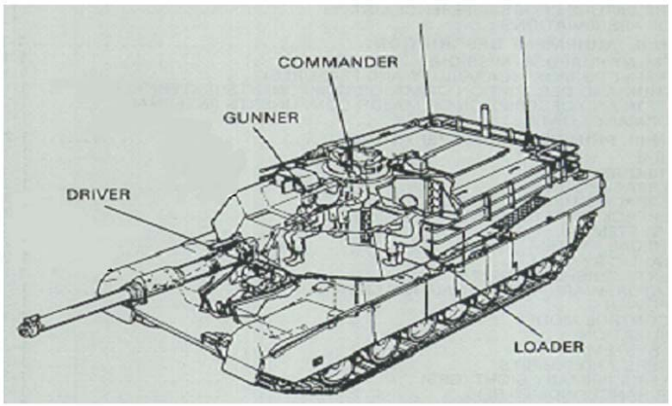

Fig.2. Main design of the Tank

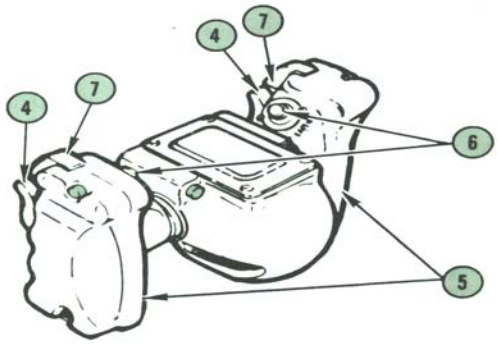

Fig.3. Gunner's Power Control Handles

Fig.4. Gunner's primary sight symbology

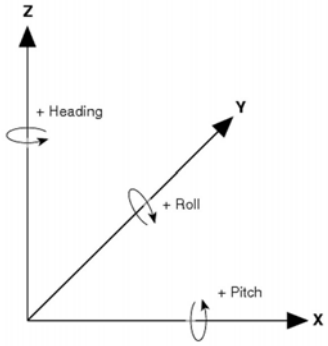

Fig.5. Right-Handed Coordinate System

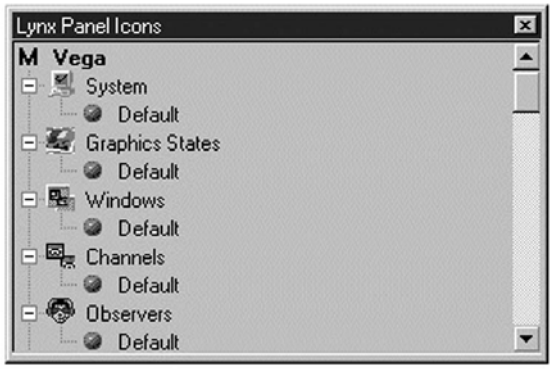

Figure.6. LynX Tree

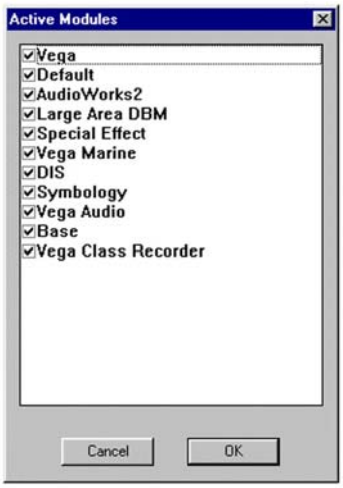

Fig.7. Active Modules Dialog Box

```
#include <vg.h>                    /* main include
file for Vega    */
#include <pf.h>                   /* main include
file for Performer    */
#include <update.h>
void main ()
{
   /* init, define, and config the system */
   vgInitSys();
   vgDefineSys(name.adf);
   vgConfigSys();
        while ( 1 ) {                /* forever */
                vgSyncFrame ();
                vgFrame ();
                update();
   } }
```

Fig.8. A simple Application program

Fig.9. The simulator trainer view

Table 1. some functions of gunner's primary sight panels

Control	**Function**
FLTR/CLEAR/SHTR	Positions filter, clear window, or shutter in gunner's primary sight day switch optic system.
GUN SELECT switch	Selects main gun or coaxial machine gun firing circuit for firing, or trigger safe so neither gun will fire. Resets to safe when power is turned off.
MAGNIFICATION lever	Selects optical 3X or 10X magnification for GPS day optical system.

APPENDIX.A

steps for creating an ADF file

To create our application definition, we began with the default ADF and added instances to the classes. Some classes are optional and some required. Invoking lynx without arguments displays a default ADF containing instances of all the required classes plus a few other instances for convenience.

1-Invoke LynX from the Start menu or by typing lynx at the command line.

2-Go to the following panels and note that instances, in each case called Default, have been created: Windows, Channels, Observers, Scenes, Lights, Applications, Motion Models. Before we run our application with this basic ADF, we must add a few items. The Default scene does not contain any objects. We must not only add objects to the Objects panel, but we must also put some of those objects into the scene that is referenced by the observer. If we want To move around the scene or to have objects in the scene move, we must relate a motion model to an observer or to a player.

3-Start the Active Preview tool to see changes to the application definition as they are made.

4- In the Objects panel, add an object to the list by using the Objects menu item New... Use the ? (file browser) to find the object file to associate with this object. Using the file browser to enter the object file name assures that the directory containing the file is entered into the Paths panel.

5-Position the object in the scene.

6-In the Scenes panel, add the object to the scene.

7-Use the Active Preview tool to position the object where you want it. The object is only visible in the Active Preview window after the object has been added to the scene.

8-In the Motion Models panel, select the Motion Model Type, and the Initial Reset Position for the default motion model. See The Motion Models Panel for a description of the basic motion models that LynX provides.

9-Select the Observers panel and verify that the scene containing the object is selected in the Scene drop list.

10-In the Positioning Method section of the panel, choose the Motion Model item in the options menu, and select the name of the motion model.

11-Use the Active Preview tool to try out the various motion model parameters. Remember that in this example, the observer is moving, not the object.

12-Exit the Active Preview tool.

13-we can control the whole features of our objects by using Application Program and call ADF file.