Compact Terrain Database Application Programmer's Interface For The IST CGF Testbed: Final Report

1-1-1997

S. Tan

Find similar works at: http://stars.library.ucf.edu/istlibrary

University of Central Florida Libraries http://library.ucf.edu

Recommended Citation


This Research Report is brought to you for free and open access by the Digital Collections at STARS. It has been accepted for inclusion in Institute for Simulation and Training by an authorized administrator of STARS. For more information, please contact lee.dotson@ucf.edu.
Purchase Order Number 0625971
Simulation Technologies, Inc.
111 West First Street, Suite 748
Dayton, OH 45402
25 September 1997

Final Report
Compact Terrain Database Application
Programmer's Interface for the IST CGF Testbed

Institute for Simulation and Training
3280 Progress Drive
Orland, FL 32826

University of Central Florida
Division of Sponsored Research
Final Report

Compact Terrain Database
Application Programmer's Interface
for the
IST CGF Testbed

IST-CR-97-27
25 September, 1997

Prepared for:
Purchase Order Number 0625971
Simulation Technologies, Inc.
111 West First Street, Suite 748
Dayton, OH 45402
25 September 1997

Prepared by:
S. Tan

Reviewed by:
Robert Franceschini
## ACRONYMS

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D</td>
<td>Two Dimensional</td>
</tr>
<tr>
<td>3D</td>
<td>Three Dimensional</td>
</tr>
<tr>
<td>API</td>
<td>Application Programmer's Interface</td>
</tr>
<tr>
<td>CGF</td>
<td>Computer Generated Forces</td>
</tr>
<tr>
<td>CTDB</td>
<td>Compact Terrain Database</td>
</tr>
<tr>
<td>DEC</td>
<td>Digital Equipment Corporation</td>
</tr>
<tr>
<td>DIS</td>
<td>Distributed Interactive Simulation</td>
</tr>
<tr>
<td>HLA</td>
<td>High Level Architecture</td>
</tr>
<tr>
<td>IST</td>
<td>Institute for Simulation and Training</td>
</tr>
<tr>
<td>LOS</td>
<td>Line of Sight</td>
</tr>
<tr>
<td>ModSAF</td>
<td>Modular Semi-Automated Forces</td>
</tr>
<tr>
<td>NaN</td>
<td>Not A Number</td>
</tr>
<tr>
<td>PC</td>
<td>Personal Computer</td>
</tr>
<tr>
<td>SIMNET</td>
<td>Simulator Networking</td>
</tr>
<tr>
<td>PVD</td>
<td>Plan View Display</td>
</tr>
</tbody>
</table>
1. Introduction

This document was prepared by the Institute for Simulation and Training (IST) for Simulation Technologies, Inc. (STI). It is a required deliverable under P.O. #0625971, “Compact Terrain Database Application Programmer’s Interface Upgrade for the IST CGF Testbed.”

The IST Computer Generated Forces (CGF) system is a simulation application used for Simulator Networking (SIMNET) and Distributed Interactive Simulation (DIS) exercises, as well as for research in several areas of CGF. The IST CGF Testbed version developed in this project runs under MS-DOS and is compiled with the WATCOM C/C++ v10.0 compiler to create a 32-bit DOS4GW executable for IBM-compatible Personal Computers (PC). It provides both 2D (plan view) and 3D (wireframe) visual displays.

This project developed a version of the IST CGF Testbed capable of using terrain databases in Compact Terrain Database (CTDB) format version 7. This is a continuation of a previous project entitled “CTDB Conversion of the IST Computer Generated Forces” under STI Contract No. PO0411966. The previous project developed an API to allow the IST CGF Testbed to work with CTDB v.5. In brief, the basic approach was to map IST CGF Testbed terrain functions to CTDB API calls; the details were covered in the final report from the previous project and are not repeated here.

This report describes the effort of enhancing the IST CGF Testbed’s terrain interface to work with CTDB version 7. This effort includes an upgrade to allow the IST CGF Testbed to process data from CTDB version 7 files, and limited testing of that upgrade on some terrain databases commonly supplied with ModSAF. This effort does not include development of new CGF capabilities to make use of the new features of the CTDB version 7.

2. Approach

The approach to accomplish the current task is to use the same conversion technique as described in the previous CTDB conversion project, making similar changes to the sources and building the final IST CGF Testbed executable as a result. By adopting this approach, the whole conversion process can be broken into the following three major steps:

1. Changing the source and header files, compiling and building the library libctdb.lib. This is the library containing all the CTDB API modules to be used by the IST CGF Testbed.
2. Changing the IST CGF Testbed source and header files, compiling and building the IST CGF Testbed executable.
3. Testing the IST CGF Testbed executable using CTDB v.7 terrain databases.

The above has resulted in a functional version of the IST CGF Testbed, compiled with the WATCOM C/C++ compiler, capable of utilizing CTDB databases v.7.
3. Changing, Compiling, and Building the CTDB Library

The first step in the process of modifying the IST CGF to read CTDB databases was to compile the CTDB API source, as available in the ModSAF distribution, using the WATCOM C/C++ v10.0 compiler. The development was performed on an IBM compatible PC using MS-DOS Version 6.22.

3.1 Changing file names

First, the MODSAF CTDB related source and header files were transferred from the UNIX machine to the PC. Since DOS on the PC does not allow long file names, they were renamed to the shorter 8 character file names to be used on DOS. A total of 26 header files and 58 source files were transferred to the PC and placed in the cgf/modsaf sub-directory of the IST CGF Testbed’s source code. This is where the library file libctdb.lib will be built. The files are listed as follows:

address.h  basic.h  ctextport.h  cttfeat.h
ctpost.h   libbasic.h  libclass.h  libctdb.lib
libctdb.h   libgcs.h   libgcs1.h   libreade.h
libvecma.h  poconst.h  ppo.h      stdalloc.h
stdbcopy.h  stdcoord.h  stdext.h   stdroute.h
stdstri.h   stdstrin.h  stdtypes.h  texport.h
vmatmatm.h  vmattran.h  ascii.c    ct08bitt.c
ct16bitt.c  ctaccess.c  ctappear.c  ctbridge.c
ctcache.c   ctclip.c    ctcontou.c  ctdpbuff.c
ctdqbuff.c  ctdtc.c     ctelev.c    ctencode.c
cfteat.c    cttgi.c     cthg.c      cttow.c
ctinit.c    ctinterv.c  ctlookup.c  ctmes.c
ctmesbox.c  ctmicro.c   ctmlib.c    ctmr.c
ctplace.c   ctpoly.c    ctprint.c   cttprof.c
cttpop.c    cttquad.c   ctraster.c  cttroute.c
cttsearch.c  cttopo.c   ctveh.c     gcscasses.c
gcsconv.c   gcsinit.c   gcssutil.c  gcsvect.c
header.c    posts.c    vmatadd.c   vmatangl.c
vmatassm.c  vmatcros.c  vmatdot.c   vmatmatm.c
vmatnega.c  vmatorie.c  vmatproj.c  vmatrota.c
vmatscal.c  vmattran.c  vmatunit.c  vmatvecm.c

It is to be noted that ctmes.c and ctmesbox.c are two new files for CTDB v.7 not previously present for CTDB v.5. So, the two files were added to the makefile to allow their compilation.

Also, in core.mak, reference to $(CORE)\modsaf21 has been changed to $(CORE)\modsaf. In common.lnk, reference to ..\core\modsaf21\libctdb.lib has been changed to ..\core\modsaf\libctdb.lib
3.2 Changing CTDB API Sources

The following is a summary of the modifications which have been made to the MODSAF CTDB API source and header files.

1. On the UNIX machine, most of header files are included into other source or header files using the #include <filename> directive. Since all the CTDB source and header files on the PC are located in the cgt\modsaf sub-directory, they were changed to use the #include "filename" directive instead.

2. The CTDB API header files were modified to reflect the big endian/small endian compiler discrepancies. This is particularly important for the allocation of space for bit-fields in structures. Most of the UNIX compilers tend to allocate space more minimally (more compactly) for a bit-field in a structure than the WATCOM compiler. For example, if unsigned int is used in a structure to define a bit field for 6 bits, most UNIX compilers will allocate 8 bits for the entire structure, while the WATCOM compiler will allocate a full 32 bits for the same bit-field structure. To force the WATCOM compiler to more efficiently compact the bit-field structure, one uses unsigned char to cause 8-bit allocation. Similarly, one can use unsigned short to force 16-bit allocation. The rule of thumb is to use the smallest of unsigned char, unsigned short, or unsigned int as appropriate to the bit-field definition in the structure to yield the most compactness. Most of the big endian/small endian compiler discrepancy changes were done in ppo.h.

3. All references to the function fsqrt() have been changed to sqrt() (ctappear.c ctbridge.c ctgi.c cthtw.c ctlookup.c ctmgco.c ctmr.c ctpoly.c gesconv.c gsuseil.c gcsvec.c vmatangl.c vmatorte.c)

4. All references to the function random() have been changed to rand() (ctcache.c)

5. A bug (an extraneous ";") was found in ctmes.c in the function ctdb_mes_to_xyo and was removed as shown below:

The following statement in the function:

if ( gcscell != mes_cell );

has been changed to:

if ( gcscell != mes_cell )

6. In ctinit.c:

change the line

if (!fopen(fname,"r"))

to

if (!fopen(fname,"rb"))

7. In ctinit.c:
change the line
    sprintf(filename,"%s/%s.%s%d%s",tdb_path,dbname,
  to
    sprintf(filename,"%s\%s.%s%d%s",tdb_path,dbname,
because DOS uses \ for pathname and filename separator.

After the changes were made to the CTDB sources, we invoked the make command to initiate the compilation. The result of the compilation was the creation of libcctdb.lib in cgf\core\modsaf sub-directory.

### 3.3 Changing, Compiling, and Building the IST CGF Testbed

Next, we edited and compiled the IST CGF Testbed sources. The IST CGF Testbed sources that reference the CTDB terrain are localized in just 3 files residing in cgf\terrain sub-directory. They are:

- terrain.c
- building.cpp
- dispintr.cpp

The following is a summary of changes made to the IST CGF Testbed source files:

An error message "Name 'is_sw_ne' not found in struct ctdb" was displayed when trying to compile terrain.c. It was discovered that the bit definition is_sw_ne, though previously defined for CTDB v.5, has been removed in CTDB v.7. This bit defined the direction of elevation post triangulation. In CTDB v.5, it is defined for the entire terrain database, i.e., the triangulation direction is the same for all elevation posts in the terrain. In CTDB v.7, this is defined for each individual elevation post and the bit, POST_SW_NE, is embedded in the elevation post data. To solve this problem, the function IsSWNE() has been changed to check the bit, POST_SW_NE, in the elevation post data and all references to IsSWNE() have been changed to reflect the new parameters required to be passed to IsSWNE().

```c
int IsSWNE(int32 normx, int32 normy) {
    int32 post = ctdb_lookup_elev_post(&ctdb, normx, normy);

    return (post & POST_SW_NE ? 1 : 0);
}
```

None of the other IST CGF Testbed sources and header files needed changes. The result of make is the creation of sim.exe in the sub-directory cgf\sim.

### 3.4 Testing the IST CGF Testbed simulator (sim.exe)
The following is a report of our observations when testing the IST CGF Testbed simulator on the CTDB terrain. The testing was performed on a 486 PC and a Pentium PC.

Several CTDB terrains were used for the testing as listed below:

dto_test01.c71 (2.2 MB)
hunter01.c71 (5.5 MB)
Knox03.c71 (6.4 MB)
ntc01.c71 (2.2 MB)
itsec93.c71 (10MB)
moba1196.c71 (2.1MB)

1. It is to be noted that EMM386 memory manager should not be loaded in autoexec.bat or else the IST CGF Testbed simulator program would not run.

2. When the IST CGF Testbed simulator program is started, you might get the message "failed in SetMulticastList() #6". This message should be ignored as it is unrelated to terrain testing.

3. For the purpose of testing CTDB terrain, you can disable network packet sending and receiving by setting the corresponding options in sim.lod. This is necessary if you do not have the corresponding packet driver for the network card on your machine (as is the case for our Pentium PC).

4. Try to run the IST CGF Testbed simulator in real DOS mode as opposed to a DOS window under Windows 95. We got the error message "DOS/4GW error: exception OEh (page fault), when using hunter01.c71 terrain if CGF was run in a DOS window under Windows 95. But no such problem occurred when it was run in real DOS mode. The problem might be due to incompatibility between Windows 95's memory management and DOS/4GW DOS extender used by WATCOM.

5. When attempting to run the IST CGF Testbed simulator on the Knox03.c71 terrain, we got the error message, "allocation of 243007668 bytes for MES template vertices failed". The Pentium PC for the test has only 64 MB of RAM and was thus unable to honor the allocation request. However, we were able to run the CGF simulator using the other terrain databases (dto_test.c71, ntc01.c71, hunter01.c71, itsec93.c71). We believe there is a corruption in the data for the Knox database (possibly occurring when the database was converted to little endian by using the ModSAF-supplied converter).

6. Terrain extents are shown as a rectangle on the screen. When testing with dto_test01.c71 terrain, it was observed that green lines (tree lines) extended outside the terrain extents. With printf debug statements added, the extents were found to be maxx=20625 maxy=20625 which seemed to agree with the extents rectangle shown on the screen. It was suspected that the problem might be due to the terrain database (dto_test01.c71) itself or due to the ctdb_get_extent() function which did not return the correct extents values to the IST CGF
Testbed. When repeating the same test for the other CTDB terrain databases (dto_test.c71, ntc01.c71, hunter01.c71, itsec93.c71), the problem did not occur (no tree lines displayed outside the terrain extents). So, it is more likely that dto_test01.c71 terrain might have some bad values for the terrain extents.

7. An entity was created and allowed to move randomly on the terrain. Observation was made of its movement using the IST CGF Testbed simulator's 2-D and 3-D display. The following observations were noted: Sometimes, it might display the message, "initial bounds check failed in MakeGrid()". This message is to be ignored as the random move algorithm might pick a random destination outside the terrain extents. With the vehicle randomly moving around on the terrain, its movement on the terrain was observed with the 3D display. The vehicle seemed to follow the terrain contour while moving on the terrain. (i.e., it did not seem to fly in mid-air, or drive below the ground surface). However it was observed that vehicles can drive through buildings. The IST CGF Testbed behaviors code has not been adapted to recognize the multi-elevation structures and hence allows the vehicles to drive through them.

8. When testing the upgraded CGF on the Ft. Benning terrain (mobal196.c71), it was observed that at certain locations on the terrain, the program would encounter an exception when terrain polygon drawing was activated. In particular, the location (16500, 15000) was very consistent for duplicating the problem, though the problem did not occur at most other locations. Further investigation revealed that the program was trying to call ctdb_lookup_soil to determine the soil type at that location. This is a MODSAF CTDB API function and it makes calls to other API functions nested to quite a few levels down. The problem was traced to the ctdb_lookup_feature_elevation_loop API function. It was using an invalid feature type of 6 to locate the CTDB_FEATURE_DATA structure thus causing the exception. Whether the invalid feature type of 6 was due to bad data in the terrain database or otherwise needs further investigation.