Adding a new feature to L3DGEWorld 2.3 - a guided tour through the codebase

Chris Holman*
Centre for Advanced Internet Architectures, Technical Report 121213B
Swinburne University of Technology
Melbourne, Australia
6963420@student.swin.edu.au

***Abstract*—This report gives a brief tutorial on the L3DGEWorld 2.3 code base. We provide some example code to add extra attributes and entities to L3DGEWorld and then discuss the major locations of the L3DGEWorld code within the OpenArena code.**

***Index Terms*—CAIA, L3DGEWorld, source, code**

I. Introduction

L3DGEWorld was developed on top of OpenArena. OA uses a modified version of the idTech 3 engine, which was created for Quake III Arnea. The engine is already documented in various places [1]–[3], so we will focus on L3DGEWorld specific code. This report aims to provide a brief guide to the structure of L3DGEWorld 2.3's code by providing a few tutorials to modify its behavior, as well as a few notes about where to find key parts of the L3DGEWorld code.

In 2005, the source code for the idTech 3 engine was released under a GPLv2 license [4]. From this, two key projects were spawned: ioQuake3 [5], which continued to improve and expand upon the idTech 3 engine; and OpenArena [6], which utilises ioQuake3 and created its own GPLv2 licensed maps, models and other assets. L3DGEWorld is a modification of these projects.

We reference the source code from L3DGEWorld 2.3 [7], which uses OpenArena 0.6.0, including ioQuake3 revision 982. Our test development environment was Windows 7 with MingW32 [8].

For information on other aspects of L3DGEWorld, see the L3DGEWorld Papers and Interim Results page [9].

The remainder of the report is structured as follows: section II discusses some simple modifications that can be made to L3DGEWorld's source code, section III lists where the major sections of the L3DGEWorld code modifications lie, relative to the ioQuake3 source code, and section IV concludes the report.

II. Modifying L3DGEWorld

The L3DGEWorld 2.3 source code is available inside the L3DGEWorld 2.3 package [7], inside the /source/ioq3-l3dgeworld.tgz archive. Instructions to compile L3DGEWorld 2.3 are in /source/README.txt.

Our modifications to the code focus on entities and their attributes[1]. In L3DGEWorld, an entity is one object with a defined set of attributes. Each entity is represented by a single model, such as a star, phone or switch, inside the L3DGEWorld environment. The attributes define how the entity behaves - bounce height, spin rate and colour are three of the 9 attributes.

A. Warning!

The idTech 3 engine has a number of hard coded limits in place. These limits can be modified - whether or not you will experience any adverse effects depends on a number of factors, such as how many entities you have in your map.

If you experience crashes or you cannot join a server when you were previously able to, reduce some of these constants, closer to what they originally were.

Given the changes listed in this document, a modified client will not work with an unmodified server, and vice versa.

B. Limitations

The data for the entities and attributes is stored in a giant array of configuration parameters, and every attribute value for every entity has a hard coded position in this

*The author was an undergraduate engineering student undertaking an IBL placement at the time of writing this document

[1]Attributes were previously called 'metrics'. The code still refers to metrics, while this document uses attributes. The two terms are interchangable.

array, as defined by \code\game\bg_public.h, lines 103 to 111.

By increasing various constants such as MAX_METRICS or MAX_L3DGEHOSTS, we have increased the number of entries that need to exist in this array. If the number exceeds MAX_CONFIGSTRINGS, we must increase MAX_CONFIGSTRINGS such that they can all fit.

```
\code\qcommon\q_shared.h:
928: #define MAX_CONFIGSTRINGS
21,000
```

Increasing this too far will result in the game crashing when it attempts to join the server. We were able to increase it to 21,000 before crashing occurred - double its original value.

C. Adding new attributes

A colleague wanted to modify the X/Y/Z scaling of an entity independently of each other dimension, unlike the existing scale attribute which scales all at once. We use this as a practical modification example.

The number of attributes in L3DGEWorld is defined as a C constant in the file \code\qcommon\l3dge.h.

```
#define MAX_METRICS 10
```

Increasing this increases the number of attributes. As per Section II-B, the attribute values are stored in the configuration array. If this array fills, you will receive an error at compile time advising as such and you should increase MAX_CONFIGSTRINGS. See Section II-B for how to do this.

In \code\cgame\cg_ents.c exists the function CG_Item(). This function is called for each in-game entity every tick. One of the functions performed by this function is applying attributes to entities. On line 349 is:

```
if (item->giTag == HI_L3DGEHOST) {
```

Within this if block, all of the current l3dge attributes are applied. To add our new attributes, we will add our code in here.

First, declare a new float, one for each new attribute:

```
float metric10=0;
```

Then, retrieve the value for each new attribute in the same manner as the original metrics:

```
metricstring = CG_ConfigString(
L3DGE_METRICRATE( 10,
cent->currentState.hostid));
metric10=atof(metricstring) /10;
```

Do this for each dimension, creating metric 11 and 12 accordingly.

Entity scale changes are smoothed over a period of time, instead of being an instant change. The code to implement this feature is between line 431 and 474. Applying this visual nicety to each of the three dimensions is left as an exercise for the reader. The centity_t struct (\code\cgame\cg_local.h line 170) will need to be modified to store state for each dimension between ticks.

On line 620 to 622, the three dimensions are scaled using VectorScale(). The second argument is the factor to scale by. Modifying these three lines as below will complete the necessary changes.

```
VectorScale( ent.axis[0],
cent->metric10, ent.axis[0] );
VectorScale( ent.axis[1],
cent->metric11, ent.axis[1] );
VectorScale( ent.axis[2],
cent->metric12, ent.axis[2] );
```

Note that the ioQuake3 engine uses currentScale to determine if an entity is still within the player's field of view. If any of the individual dimension scales are larger than currentScale, the ioQuake3 engine may decide that it should not render the entity while the entity is still visible, resulting in the entity suddenly disappearing from view.

D. Increasing the number of entities

```
\code\qcommon\l3dge.h
#define MAX_L3DGEHOSTS 256
```

You will need to create or modify an existing map with this many entities. The largest default map has around 140 entities.

There are other limits in place that require further investigation, as only 255 entities will be in the environment. Interestingly, the 255th does not have a label appear above it.

III. Useful code sections

qcommon\l3dge.h

Contains constants relating to L3DGEWorld - notably the entity limit and the branding text.

server\sv_main.c

Contains SV_ConnectionlessPacket() - the function which chooses how to process an incoming L3DGEWorld daemon message. All L3DGEWorld packets are considered connectionless by the Quake 3 engine as they all begin with the Quake 3 connectionless packet identifier, 0xFFFFFFFF.

server\sv_l3dge.c

Contains most of the L3DGEWorld specific code, including functions to handle token generation and negotiation, sending and receiving daemon update messages and such.

SVC_L3DGEInput is of particular interest, in how it processes the input l3dge messages.

```
Cmd_ExecuteString (va("cvarset
%s",split));
```

`split` is tilde separated set of values to set a property of an entity - eg, `~001~1~r~40~`. This is the tilde separated string that is sent from a l3dgecomm daemon to update an attribute.

`cvarset` is a custom command, only used by L3DGEWorld-specific code. It calls `SV_CVARSet_f` in `server\sv_l3dge.c`. This then calls `parseL3DGECommand`, which inserts the values in to the engine ConfigString, as seen in Section II-B.

Because Cmd_ExecuteString() is used to update attributes in this manner, the same command can be used from the console to alter attributes. This means that there is no need for a l3dgecomm daemon if you only need to see what something simple looks like. First, copy a ~ character to your clipboard - ~ is used to toggle the console, so you cannot type it while in game. Then, toggle the console and type: `cvarset ~001~3~r~40~`. Use ctrl+v to paste the ~ symbol instead of typing it. Entity 1 should now be spinning very fast.

cgame\cg_ents.c

Contains CG_Item() - as per section II-C, this function is what is used to apply attributes to entities. It is called on every entity once per engine 'tick'. The renderer uses information calculated in this function to determine how to render each entity.

cgame\cg_local.h

Defines the centity_t struct, which is what each entity is stored as. L3DGEWorld stores a number of state variables in this struct, including bounce height and rate, scale and the direction an entity is facing.

q3_ui\ui_l3dgemessages.c

The details UI that appears when the "inspector" tool is used is implemented here.

IV. Conclusions

In this report, we have briefly discussed the L3DGEWorld 2.3 codebase, including the major modifications made to ioQuake3 to enable L3DGEWorld functionality, and some example code to add extra entities and attributes to L3DGEWorld.

References

[1] D. Stefyn, A. Cricenti, and P. Branch, "Quake III Arena Game Structures," Tech. Rep. 110209A, Centre for Advanced Internet Architectures, Swinburne University of Technology, Melbourne, Australia, 09 February 2011.

[2] F. Sanglard, "Quake 3 source code review." http://fabiensanglard.net/quake3/index.php. Accessed Nov 14th, 2012.

[3] Prometheum, "Looking at the quake 3 source – part 1." http://element61.blogspot.com.au/2005/08/looking-at-quake-3-source-part-1.html. Accessed Nov 14th, 2012.

[4] id software, "id-software/quake-iii-arena." https://github.com/id-Software/Quake-III-Arena. Accessed Nov 14th, 2012.

[5] ioQuake3, "ioquake3." http://www.ioquake3.org. Accessed Nov 14th, 2012.

[6] OpenArena, "Openarena." http://www.openarena.ws. Accessed Nov 14th, 2012.

[7] CAIA, "L3dge - downloads." http://caia.swin.edu.au/urp/l3dge/download.html. Accessed Nov 14th, 2012.

[8] "Mingw | minimalist gnu for windows." http://www.mingw.org/. Accessed Nov 14th, 2012.

[9] CAIA, "L3dge - papers and interim results." http://caia.swin.edu.au/urp/l3dge/papers.html. Accessed Nov 14th, 2012.