FEEDER

report for the special course at IMM, DTU
Game Production

20th March 2007

Jakub Cupisz

1 Preface

This report describes the work I have made during DADIU's March production 2007. You will find here information on jobs assigned to me, problems that I have encountered and some technical info about Source engine.
Our headquarters were based at IT-University.

2 Assignments

2.1 Setting up production pipeline

We have spent the first day on setting up programs that we will use throughout the production. First Steam, then downloading and installing Half-Life 2. Then we have set up SVN server on on of the computers. Also Maya and 3D Studio Max have been installed.

2.2 To Fixed View

Camera setting 1
The camera behaves like a standard FPS camera controlled by the mouse and keyboard.
Used when the player searches for food.
Camera setting 2
The camera locks into position and the player can use the mouse for different purposes.
Used when the player is feeding.

The player switches between camera settings by clicking the CTRL button.
Setting 1 can be always be activated, but setting 2 is only applicable if the avatar is standing on certain hotspots on the level (in front of the gainer).

Camera setting 1, first person view, is the default option in Source Engine. So what needed to be made, was a transition to setting 2 (fixed view).
First I have made a Logical Entity (*FixedViewEntity*). *It's an object that exists to service input from the game map and make decisions based on the state of the world.*
In particular:

- it gets input when a user collides with a trigger - this enables CTRL button to be pressed
- CTRL button is bound with entity's *CtrlPressed* input function

When the user presses CTRL, gainer is localized on the scene, final position and orientation is being used to calculate total translation and rotation. Then a *Think* function is started being called every 0.02 second. *Entities have the ability to update internal state and make decisions via a think function.* In every call a small step is being added to translation and rotation. When the current translation and rotation is closer then EPS to the final values, calling Think function is abandoned, and the final values are set. Thanks to this step, the contents of the frame in fixed view position are always the same.

2.3 Right hand

2.3.1 Importing models

Since noone from our group knew how to import animations from Maya, I have to made a research in this topic. It took a while to figure out how to set up folders contents, paths in the exporter, and sequences parameters within a model file. Also applying a bump map was not obvious - bump map object has to have a „_normal” name to be exported properly.

Animator seeing how many steps you need to apply texture, give correct names and set up paths - denied doing that. He said there was no way he will manage to do that. Since I didn't have time to argue with him and I was aware, that he could run into problems with it - I spend one evening just on exporting models.

2.3.2 Idea

In this control scheme the mouse movement should be more or less equal to the avatar's right hand movements (the gesturing hand). If the player moves the mouse gently up and down, the gainer will experience a caress. If the player moves the mouse quickly from right to left the gainer will be slapped.

2.3.3 Bones animation, morphing?

There were two approaches to the problem. First - using bone's, inverse kinematics and such. Second - using precisely defined animations and figuring out how to make transitions between them.
The first topic is quite vague for us, and it would cause more work for us than for the animators. And since we wanted to make the game playable as soon as possible and not to run into problems with time at the end, we have chosen the second approach.

2.3.4 Common frame

One problem still remains - what to do, when a user is caressing, has a hand on top of the gainer's head, and suddenly moves the mouse left or right? The hand is outstretched, and in slapping it is first closer to the player's eyes. How to make a transition between those poses.
One might think of making a blend between those two. But since it doesn't make sense for us, to allow the player to hit the gainer on the top of the head, we would have to add some constraints. A much simpler idea is to make a common frame for all the animations, and make transitions to always go through this frame. Using only predefined animations will constrain the user automatically, but in this case - it is what we want.

Animator responsible for hands has made all animations of the right hand with a common frame. Animations, that I had to handle were:

- from idle to common - played when user enters feeding mode
- slapping - common frame is somewhere in the middle of the animation
- caressing - common frame is the starting point for the animation
    - from common the head area
    - contact with the head

We have divided caressing animation into two parts: upward movement of the mouse controls the first one - from common frame to area close to the head, and then - up / down movements control the hand only in that area. Movement right, getting the hand off the face, is needed to go back to the common frame. This actually gives a better user experience, since the player is not confused by the hand going off the face when he still wants to caress.

Moving up in slapping animation, will cause the animation to go to the common frame, and then proceed with caressing.

This part required a lot of tweaking, since decisions of transitions had to be made based on mouse delta values, current state, and current point in the particular animation. But overall - the code combined with the animations worked out well. Players who have tested the game didn't notice any failures in the hand movement. Everything just felt right and natural.

2.4 Left hand

Left hand is responsible for:

- picking up things and dropping them when walking around the kitchen
- feeding the gainer with the food currently in the hand

2.4.1 Eatable Objects

Eatable objects are Model Entities. They have a visual component and can move around the map. We have derived it's class from CphysicsProp to gain physics calculations automatically - useful when dropping items.
Eatable object have couple of properties: type, size, calories number (increases stomach meter), grabbing type (defines the animation used for showing the object in front of the player).

2.4.2 Picking

When the user points on an object and left clicks, the object should appear in his inventory.
Based on the player's position and viewing angles, I am using *UTIL_TraceLine* to spot entity laying below the crosshair. Then I am using newly defined collision group to determine the type of the object.

Picking animation is played next. When the hand is in fully extended position - the object is attached to one of the hand's bones, and continues moving with it away from the screen. During the whole animation the alpha value of the model is being decreased.

2.4.3 Inventory

Left hand entity stores the information of the objects that have been picked up. This information has to be passed to HUD, which is client side. I have used User Messages and WRITE_FLOAT() to transfer the data to HUD, where Thomas' code used it to choose proper icon and show it on the screen.

User can then use buttons from 1 to 7 to select an object from the inventory or simply scroll the mouse wheel to cycle through them. It causes another animation to play - from holding the object outside of the screen to having it in front of the player.

Problem: when having the object in the hand and moving around, the object sometimes disappeared, showed up in the initial lying position and wobbled there.
Solution: when setting the object's position and orientation, the values were only set for the model, not for the underlying physical model. ent->VPhysicsGetObject()->SetPosition(...) did the trick.

Also situations when the user cycles fast through the inventory, picks an object when holding another one in the hand, or picks and object when picking animation for another object is being already played, have been solved.

3 Technical

3.1 ConCommands

Since compilation, linking and running the game takes about 5 - 7 minutes even for changes in highly isolated code, it has even more sense to make a mechanizm for tweaking parameters from within the game's developer console.
Console commands usually pass the parameters to some code function and execute it, but they don't remember these parameters.
I have made console commands for:

- changing the hands position offset from the viewing origin / direction
- adjusting the position of the „hot spot" on the floor in front of the gainer
- setting player's eye height -also an accessor had to be added to player's class
- tweaking hands parameters:
  - sensitivity of the mouse
  - head area boundaries - for caressing
  - epsilon values for determining switches between different types of states
  -

3.2 Client Server Communication

This type of communication between server, which is responsible for game logic, and client, which in general draws the scene and gets user input, is obviously necessary.
One of the uses for Server -> Client communication was to transfer the information that the view is fixed from the *FixedViewEntity*, which is server side, to the *input* object responsible for reacting to user input. I have used engine->ServerCommand(conCommandString) with the console command name registered client side. It's worth noting, that conCommandString has to end with *\n*.

When in fixed view, *input* object should not update player's viewing angles. It should pass mouse delta values to the right hand directly. Thanks to that - view can stay fixed, and right hand new position in the animation can be calculated.
Since right hand is a server side entity, we needed Client -> Server communication. It was realized by using engine->ServerCmd(conCommandString) with the console command name registered server side.

3.3 input and output functions

You can define input and output functions for an entity. Then, in Hammer World Editor, you can bind output of one entity with input of another one. I have used this mechanism when passing the information from the trigger object to the *FixedViewEntity*, which enabled CTRL button to be pressed.

4 Trivia

4.1 Different hand models - different origins

Slapping animation for right hand had different origin than caressing animation.
Solution:
Units in one scene in Maya file were set to inches, in another - to cm, and since the hand was not in the world origin (0, 0, 0), the position changed. The size of the hand was also different, but that was actually hard to notice, because the position was different at the same time.

4.2 Lights' ID's

During the night one day before the deadline everything fell apart. Since I live far away from the IT-University, I have gone home earlier then other fellow programmers. I got a message about the breakdown during the night.
Right in the morning I have tried calling other guys, but there were still in deep sleep. I have called the producer and bombarded him with questions. It took some time until he realized who is talking to him. He has also worked till late. Then it occurred, that he had a dream about Egypt, and when I called him, he thought it's ambassador of Egypt calling ;)
Anyways - when I got to our headquarters, the game did not run. After couple of minutes of investigating what have they changed last, it occurred that they have added some lights to the map. Removing those lights helped instantly. Why? Because they have copied them manually to the scene file from another local file. Since every object has to have an unique ID, game launch failed. ID's were not unique any more. No warning when compiling the map, nothing! Just a game crash.

4.3 Debug ->Release = crash

It's the last working day before presentation. Last things to polish in the game. Compilation in Release HL2 mode. Game crashes after 10 seconds.
The problem seems to be in the module responsible for face posing. Not in our part. Klaus tries to isolate the bug. He reverts the whole face posing related code to the original one. *Crash*. He starts from scratch just with face posing. *Works*. With his blending several animations - still *works*. Adding our another entities - *crash*. They should not have a direct influence. All what they change, is that their *Think* functions are being called multiple times per second.
Timing problem? Maybe - unsolvable in such a short time.

The vision of delivering the game in debug mode makes us unhappy.
I have tried various options for code optimization and recompiled. That made things even worse. One last try - compiling in Visual Studio 2003 (we have used 2005 version in the headquarters). Luckily I had my computer with VS 2003 that day. Setting up the mod, checking out from the SVN repository, compilation. That all takes time and makes the computer to use battery extensively. And I have no AC adapter with me.

It *works*! Release HL2 mode works when compiling under VS 2003 perfectly. 50 minutes left on the battery, and guys want to tweak something more in the code. I am leaving computer to them and going to sleep. They have managed to make two more builds overnight! Couple of minutes left on battery in the morning ;)

5 Summary

I have really enjoyed the project. Working with people from different universities, with different qualifications, but sharing the same passion - games, was a great experience. The structure of the group, based on the one used in the industry also worked well.
Programming for such a big engine for the first time broadened my horizons. It also helped me to see flaws in engine I am developing with a friend for VR course.

It was a hard work from dawn till dusk (usually much longer). But seeing happy faces of the people playing our game in the end was more then enough to compensate our efforts.