

02563 Real-Time Graphics for Games and  
Virtual Reality F08  
DADIU 2008 May production report

Robert Cupisz s070924

June 30, 2008

## 1 Introduction – Lead Programmer’s Duties



Figure 1: Game’s welcome screen with one of the more challenging tunnel sections in the background.

The role I was playing during the DADIU 2008 May production was the Lead Programmer for the game Eskimotion.

My primary duties were to assess, if it was feasible to get game features requested by the Game Designer to work in the Source engine. Later, to

specify sets of programming tasks oriented to develop those features, assess workload, divide them between myself and the two other programmers, explaining what exactly has to be done and often - how, and finally - to do the programming itself.

Additionally all the programmers had to aid the other members of the team in solving all the technical issues: general (e.g. how to use the SVN) and Source-related (e.g. why hammer doesn't load my model/texture/animation correctly?).

## 2 Design issues

The major issue during the production from my perspective, was that we failed to correctly assess (or should I rather say: guess), that some of the key game features are impossible or very hard to achieve in Source.



Figure 2: Chiku falling down after he hits an obstacle, with ice obstacle gibs flying around him.

As an example: on any non-box shaped map (Eskimotion is played in round tunnels) all sorts of bugs related to player movement and collision detection start to manifest themselves, which is exceptionally unfortunate, as the player needs to have good control over his avatar in this game. It's even nearly impossible to build a tunnel-shaped map that would compile correctly in Hammer, the level editor for the Source engine.

### 3 Programming challenge – a case study

An example of the requirements was that Chiku – the player’s avatar – was supposed to run down the tunnel all the time by itself (player controls only it’s sideways movement and jumping) and that he’s always oriented more or less perpendicular to the surface he’s running on (so up side down, when on the ceiling).

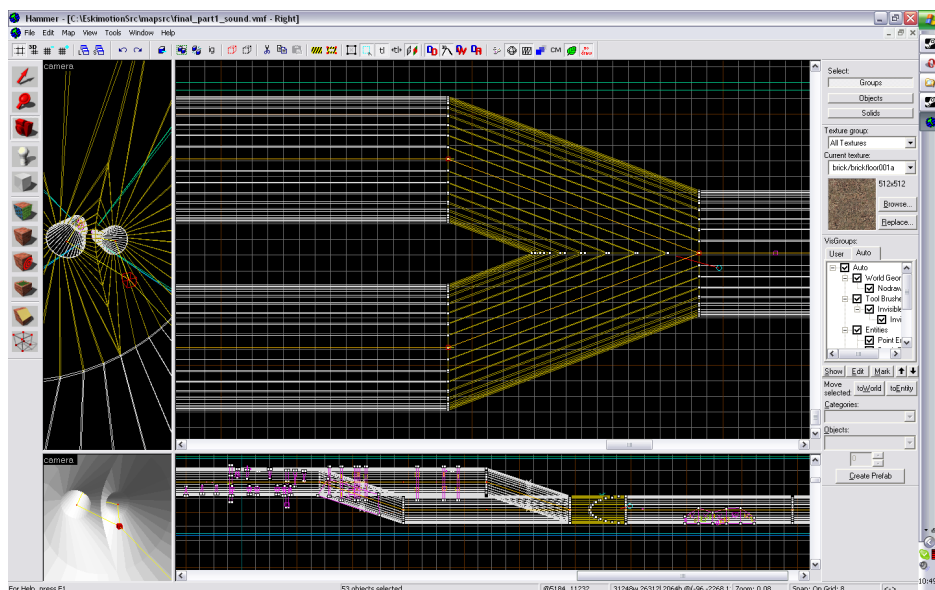


Figure 3: Top view of the tunnel in Hammer, Source’s level editor. Path is shown as red nodes connected by yellow edges.

I had a number of solution ideas. For propelling Chiku forward analysing the tunnel’s geometry was out of the question, but if the tunnel was tilted downwards and it’s surface slippery enough, Source’s physics engine should make Chiku slide correctly all the way down. While testing this concept it turned out, that player movement is never handled by the physics engine (except while in vehicles, but that’s a different case). The player movement code uses a different approach and couldn’t be modified to give results we were after.

For orienting Chiku perpendicular to the surface I could query orientation of the surface he was currently touching. This solution would have two important downfalls: a. Chiku would tilt rapidly (flicker) when running on a surface with bumps (the remedy would be to make him turn at some finite speed) and b. it would be very hard to predict how Chiku should rotate in the air to finally land on his feet.

The solution I finally applied solved both the propelling and rotating issues. I had the level designer add a path of `path_track` entities running through the middle of all the tunnels, splitting and joining where it was necessary. Path section closest to the Chiku's current location defined both the direction he should be propelled in and the axis he should be always pointing with his head at.

It turned out to be an efficient and robust solution with the obvious downfall of the level designer having to define the paths manually.

## 4 Summary



Figure 4: Chiku jumping over an obstacle. We can see that the player is performing well and he manages to keep safe distance from the snowball chasing him (shown on the HUD).

Although I do not feel I have improved my programming skills during the production, I certainly know better how to work in an unknown software environment, how to organise work for my fellow programmers and how to work with team members with no technical background.

Overall I find the DADIU production a very useful experience.