

hidden in plain sight

Gottfried Haider

Academy of Fine Arts Vienna
gottfried.haider@gmail.com

In computer systems, the translation from source code to machine code is generally hidden to the ordinary user, as are the mechanisms and properties of compiler bootstrapping [1] - with its introduction of a sort of *digital ancestry* (the function of a compiler being dependent on its own compiler). In fact, the standard way of creating a compiler toolchain for a new architecture is to cross-compile it from an *existing* compiler implementation on a different architecture. This way, a compiler program on a contemporary computer system might trace back to the origins of self-hosted compilers [2].

The aim of the art installation "hidden in plain sight" is henceforth to examine the contingencies in a seemingly static and formal environment and add this to the public discourse.

For this, a compiler program compiles itself indefinitely. Each generation then turns the (modified) source code of a popular first-person shooter game into an executable and runs it. The dynamic nature of the resulting ephemeral 3D environment is further amplified by using the live standard-output of the simultaneously running compiler process as textures for all the surfaces in the game. This way the player, which normally reverse engineers the implicit rules in the game's source code by playing [3], is confronted with a radically changed 3D environment which periodically gets `killed` and restarted as a new *world*.

References

- [1] Thompson, Ken. 1984. Reflections on Trusting Trust. *Communication of the ACM* **27**(8). New York, USA: ACM.
- [2] Hart, Timothy and Levin, Michael. No date listed. The New Compiler. *AI Memo* **39**(1). Cambridge (MA), USA: MIT.
- [3] Interview with Douglas Edric Stanley. <http://www.we-make-money-not-art.com/archives/2006/06/can-you-tell-us.php> retrieved on Feb 3, 2010.