Gianni Pucillo

Skill and experience in 3D software developing

Reverse order history:

On May 2018, a collection of algorithms and technologies has been added to the end of this history.

At the starting of 2010, *ELV* requested for an update on VisProject, looking at the incoming first *VEGA* launch, installed at the end of March. Right after the *first astounding successful* ***VEGA launch***, in February 2012 a further update has been installed with the last state of the art on computer and 3D graphic board; accelerated HIL data communication using both TCP/IP and UDP/IP protocol to interface VEGASIM (mentioned on wikipedia).

In the time, VisProject become a multi-disciplinary application without losing its own *ad-hoc algorithm* in terms of speed and performances. Now VisProject is a platform covering both ***3D scene graph engine*** *and* ***ROS***. It has been demonstrated in many application fields: home automation, installations and plants (e.g. VisPipe), CNC industrial machine projecting and use (e.g. VisMek), automated machine arms (e.g. VRobot), satellites behavior (e.g. VSat), etc.

To embrace other R&D sector where 3D was not directly available, I wrote **OpenGL/Vis.** This is an *OpenGL* interpreter developed to allow the use of *OpenGL* from within high-level applications with visual interface (such as Mathematica, Matlab, LabVIEW RT and Maple), with their own specific language. This allowed scientist to use 3D environments without knowing standard programming languages.
Meanwhile, I built a collection of functions able to perform *object reconstruction* with data received from high-tech equipment like laser beam, sonar, etcetera or software capable of vertex coordinate transmission.

Riding the last technical achievements, **VisProject** was developed. Beside the previous capabilities other were introduced: please refer to the *algorithms and technologies* at the bottom.

The evaluations around **DirectX** and **C++** porting were put on hold: DirectX wasn't available on other OSs while C++ programming on Windows was too much linked to **MFC** making the porting too much involved to the Windows proprietary OS.

In the **new century**, *Fiat Avio* accepted to port **Term3D/Vis** to *Windows NT OS* (link to the fax related to commercial agreement). The system was based on *Matrox* graphic boards while the improvements were: multi-window, double-monitor, unlimited degree of freedom (DOF), dynamic communication data packet, unlimited numbers of objects and 3D-shape in a similar-ROS structure, texture mapping, use of the *National Instruments* OBC **RT7030**, The system was successfully installed at the end of 2001.

In the **1997** *BPD Defence and Space - Fiat Avio* requests for an update, obtaining the **Term3D/2**. Hardware improvements were the new *Matrox*'s MGA graphic board and the host computer; in the software was introduced a hybrid concept: a mixed *message driven - event driven* architecture.

In the same period I did analysis and support for virtual graphic reconstruction of **Parkinson's disease** (Politecnico di Milano - Dipartimento di Meccanica). Because of the high costs, the project was dismissed.

In the **1993**, supporting the **Oncology** 3D reconstruction and isodose simulation project (Borgotrento's Hospital, Verona), I gained more experience in 3D SXCI library programming, formally based over the famous *OpenGL*: the application was demanded to reconstruct sectional radiography or nuclear-magneto resonance 3D coordinates, and to track the radioactive particle injected inside the tumor and follow up on its development (radioactivity sphere).

My skills to effort 3D software developing on PC were rapidly growing. I wrote the application **Term3D** for *BPD Defence and Space* (link to the successful installation letter) and its **VEGA space launcher** (Colleferro, Rome): the first and unique software available in Italy, capable of **space launcher full mission 3D graphic simulation**. The application was written in C language, running under DOS extender OS computer with three graphic boards: the first one for develop and control the application; the second with a high resolution 2D graphic, showing a world map image with full mission path, flight and other data, visual user interface with virtual push-buttons; the last graphic board content was designed to draw three static windows frame, inside which there were a 3D graphic animation of the full launcher body, a 3D graphic animation of a panoramic view on each booster stage nozzles, and the real-time mission path drawn inside a 3D Cartesian axes cube. All the 3D graphic objects interaction were performed upon data coming from other computer in the HIL configuration, where *Term3D* was setup. The user was able to interact with the 3D world using mouse and keyboard. This application has been successfully installed on March 1994. In this version I ideated and implemented the 32 bit parallel communication protocol.

In **1991** I started developing software for 3D graphic visualization: I was employee at *3g-electronics Matrox*, the company that realized the **first 3D graphic board for PC** along with proprietary 3D SDK called SXCI, formally based on the Silicon Graphics *OpenGL*.

Algorithms and Technologies:

- **Non limited** quantity of **DOF**, windows, tasks, project or worlds, project instance, monitor, software and hardware I/O interfaces (VisResource modules)
- **real-time** and **interactive** architecture, message and event driven environment
- **VRML** converter to proprietary file format, with optimized **compression** for project storage and data transmission
- **Four level** of 3D objects and world control commands: from simple **transposition** up to single **vertex control**, full OpenGL command (OpenGL-Vis mode)
- software **interfaces** to Mathematica, Matlab, LabVIEW RT and Maple
- VisProject & OpenGL **script** and **interpreter**
- Commands recording, playback and reverse play, data editing
- **collision** detection, direct and inverse **kinematics**, **CNC** command input and output
- **virtual camera** as object, self-defined object behavior, object and camera path definition
- most of the **OpenGL function** implementation like texture mappings, lights, colors, shading algorithm, always **without display list** support, etc.
- *utilities functions* like sound play, text messages board, etc.

professional e-mail: giannipucillo@giannipucillo.it - personal e-mail: giannipucillo@gmail.com
career website: http://www.giannipucillo.it/career - professional website: http://www.giannipucillo.it