Skills and experience in 3D software engineering

Reverse order history:

A list of algorithms and technologies has been added to the end of this history.

In 2014 I felt the need to be ready for an eventual porting of VisProject on to the web technologies. High speed optimized algorithm and hardware interfaces were granted using WebAssembly. In the use of OpenGL (that is a *de-facto standard*) I am using WebGL but especially ThreeJS.

At the following links on my *career website* I published some short screen recording about VisProject in action: ROS and Object Reconstruction.

At the start of 2010, *ELV* requested an update of VisProject, looking at the incoming first *VEGA* launch. The update was installed at the end of March, 2010. Right after the *first astounding successful* ***VEGA*** *launch*, in February, 2012, a further update was installed with the latest state of the art in the field of computer and 3D graphic board hardware; furthermore I provided an accelerated HIL data communication using both TCP/IP and UDP/IP protocols to interface with VEGASIM (mentioned on Wikipedia).

Over time, VisProject has 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), satellite behavior (e.g. VSat), etc.

To embrace other R&D sectors 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 scientists to use 3D environments without knowing any standard programming languages.
Meanwhile, I built a collection of functions able to perform *3D object reconstruction* with data received from high-tech equipment like laser beam, sonar, etc. or from software capable of 3D vertex coordinate transmission.

Riding on the wave of the last technical achievements, **VisProject** was developed. Beside the previous capabilities mentioned, others were introduced (please refer to the *algorithms and technologies* section at the bottom).

The evaluations about **DirectX** and **C++** porting were put on hold. DirectX wasn't available on other OSs while C++ programming on Windows was too closely linked to **MFC**.

In **2000**, *Fiat Avio* agreed 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 improvement was 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 **1997** *BPD Defence and Space - Fiat Avio* requested a Term3D update to the latest HW & SW technologies. Hardware improvement was the new *Matrox'*s MGA graphic board and the host computer's PCI bus interface, while the most important software improvement I designed was a hybrid concept: a mixed *message driven - event driven* architecture, in the HIL communication.

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

In **1993**, supporting the **Oncology** 3D reconstruction and isodose simulation project (Borgotrento's Hospital, Verona), I gained more experience with 3D SXCI library. The application was designed to reconstruct sectional radiography or nuclear-magneto resonance 3D coordinates, and to track the radioactivity sphere created by the radioactive particle injected inside the tumor.

My skills in 3D software development for PC were rapidly growing. I wrote the application **Term3D** for *BPD Defence and Space* (link to the successful installation letter) for 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 on DOS extender and three graphic boards: the first one for developing and controlling the application code; the second one with a high resolution 2D graphic, showing a world map image with the full mission's path and flight data, visual user interface with virtual 3D push-buttons; the last graphic board content was designed to draw three static window frames, 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 object interactions were performed upon data coming from the other computer in the HIL configuration, where *Term3D* was set up. The user was able to interact with the 3D world using mouse and keyboard. This application was successfully installed in March, 1994. In this version I created and implemented the 32 bit parallel communication protocol.

In **1991** I started developing software for 3D graphic visualization: I was an employee at *3g-electronics* (*Matrox* division), the company that designed the **first 3D graphic board for PC** along with proprietary 3D SDK called SXCI, a subset of the famous 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 levels** 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's commands **script** and **interpreter**
- commands for recording, playback and reverse playback, data editing, etc.
- **collision** detection, direct and inverse **kinematics**, **CNC** command input and output
- **camera** as object, self-defined object behavior, object and camera path definition
- VisProject does not implement use of the **OpenGL's display list**
- *Utility functions* such as sound, text messages board, windows manipulation, 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