343

N90-20659

DESIGN CONSIDERATIONS FOR A SPACE DATABASE

Lance M. Moss

Evans & Sutherland Computer Corporation
600 Komas Drive
Salt Lake City, Utah 84108

Part of the information used in a real-time simulator is stored in the visual database. This information is processed by an image generator and displayed as a real-time visual image. The database must be constructed in a specific format, and it should efficiently utilize the capacities of the image generator that it was created for. A visual simulation is crucially dependent upon the success with which the database provides visual cues and recognizable scenes. For this reason, more and more attention is being paid to the art and science of creating effective real-time visual databases. This paper investigates the database design considerations required for a space-oriented real-time simulator. Space applications often require unique designs that correspond closely to the particular image-generator hardware and visual-database-management software. Specific examples from the databases constructed for NASA and its Evans & Sutherland CT6 image generator illustrate the various design strategies used in a space-simulation environment. These database design considerations are essential for all who would create a space database.

1.0 INTRODUCTION

During 1987 and 1988, Evans & Sutherland designed and developed three databases for NASA's System Engineering Simulator. Much of the experience gained and many of the techniques used in the construction of a terrain database were transferred to the construction of the space databases, but many new challenges were encountered. This paper describes some of the challenges unique to the design of a space database and explores the techniques and strategies developed to meet these challenges.

2.0 VISUAL SYSTEM

A major part of a real-time vehicle simulator is the visual-scene-generation system. The visual system's role is to provide the visual cues that help make the simulation effective. In general, a visual system comprises a computer image generator, which is the processing hardware; displays on which the computer-generated imagery is viewed; a visual database, which is the information the image generator processes to produce images; and a set of managing software, known as the real-time system. The image generator includes a set of processors, among them viewpoint processors and channel processors. The visual database is a model of the real-world environment. The real-time system is the interactive software package that controls and manages the image generator's transfer and processing of the data in the visual database.

For a database to reach its utmost visual potential, each construct within it must be used efficiently. Therefore, no database should be designed without an understanding of its visual system's processing capabilities. Some of the necessary design information includes:

1. The configuration of the image generator (including the number of viewpoint processors and channel processors) and the update rate.
2. The maximum input and output of each processor in the image generator
3. The maximum visible output for each channel
4. The amount of time allotted for each processor to accomplish its task
5. The amount of on- and off-line storage available for the database

Database engineers must analyze all the processing requirements before a design is developed. With the resultant knowledge, a database can be constructed that will fully utilize the capabilities of the image generator. However, this information must be correlated with the specified database requirements. Sometimes database requirements tax the capabilities of the given visual system. In these cases, ingenuity in the design can satisfy the requirements and still not overload the image generator. Extreme cases may occur where specific database requirements have to be modified slightly to meet the image generator's capabilities and still provide an effective simulation.

3.0 SPACE DATABASES AND TERRAIN DATABASES

Terrain databases are usually designed at a 1:1 scale with the terrain they model and only represent a relatively small portion of the real world. A space database, on the other hand, because of the sheer size of the model environment, cannot possibly model all of its objects at a 1:1 scale. For example, one of the requirements of the NASA space databases was to create an accurate starfield. The star nearest Earth, Alpha Centauri, is approximately 4.4 light-years away. Hence, some models must be created so that they appear visually correct even though they may be mathematically incorrect.

A space database has the potential of depicting its modeled environment more realistically than a terrain database can because many of the models included in it — the Orbiter, the Space Station, the Shuttle Remote Manipulator System (SRMS), and payloads, for example — are manmade and tend to be geometrically regular. Computer graphics is an excellent medium for rendering such regularities. Natural objects such as Earth's surface and the moon can be easily and effectively modeled with photo-derived-texture patterns.

Space databases generally include less information than terrain databases. Most terrain databases cover many hundreds of square nautical miles of terrain and contain enough information to create scenes that include bushes, trees, rocks, rivers, roads, cities, and farms. But a space database must contain only a few hundred accurate stars, a brilliant sun, a realistic Earth, a life-like moon, and a few extremely high-fidelity models to create an uncanny likeness of the real thing. Figure 1 is a photograph taken of an actual display that illustrates this realism.

The space databases designed for NASA contained all these models and used only approximately one fiftieth of the file space required for a typical terrain database. Although the space databases were small with respect to their storage requirements, their visual effectiveness equalled that of any terrain database designed to date.

4.0 ORGANIZATION OF A SPACE DATABASE

A visual database is a composite data structure in the form of a tree that specifies the hierarchical relationships among the items in the database. A space database has the same format as any other type of visual database, but the design and organization of a space database are quite different from those of a terrain database.

4.1 Static and Dynamic Models

The models that compose a space database can be categorized as *static* (remaining fixed at a specified origin) or *dynamic* (having independent motion capabilities).

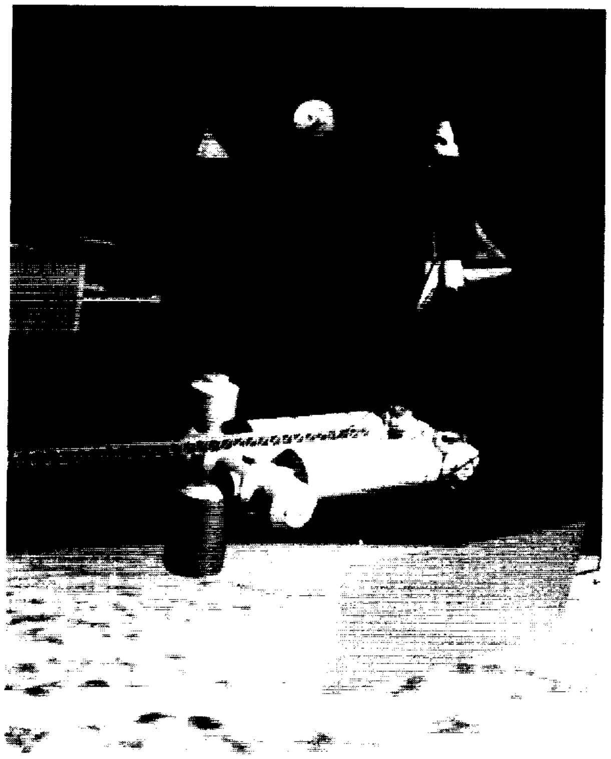
Figure 1. Photograph of a Displayed Space Database

Depending on the simulator's mission and other factors, models may switch categories. Since everything in space is dynamic, it might initially seem easiest and most logical to make all the models in a space database dynamic. However, the image generator accommodates only a finite number of dynamic coordinate systems and may not (depending on its load) be able to process all those systems at once. A static model requires very little processing time for its geometry to be displayed; a dynamic model requires additional processing time, depending upon the type of motion specified. Therefore, the number of dynamic models should generally be restricted as much as possible in order to limit the amount of time used to process them.

Depending on the particular scenario's requirements, any or all of the models of Earth, the sun, the moon, stars, or other items in the database could be static and at the same time serve in an effective simulation. For example, in a given scenario designed only to train specialists in the use of the SRMS, if the Orbiter never needed to move, it could remain static at the database's global origin. A static orbiter would have a fixed orientation with respect to the other database models, but this fact might not limit the effectiveness of the images produced for the scenario in any significant way.

The number of dynamic coordinate systems and the corresponding amount of processing time constrain the number of dynamic models that can appear in a given scenario. However, useful scenarios require that many objects (the Orbiter, the SRMS, and payloads in the space databases, for instance) be categorized as dynamic. Therefore, the database engineers must decide which models have to be dynamic for each scenario. Determining a finite list of dynamic models for a given scenario is not difficult, but if the dynamic models required for one scenario are different from the ones required for others, the complexity of the task increases significantly. When the set of scenarios requires more dynamic models than the image generator can process, an alternative must be found. The approach taken in designing and developing the space databases for NASA entailed two strategies: design of each database as a *scenario-dependent database* and a technique for choosing *model selects*.

4.2 Scenario-dependent Database

A scenario-dependent datatabase is a database that is broken into smaller, simpler pieces. Evans & Sutherland CT6 linked databases contain between 1 and 60 *entry-point sets*. Each entry-point set is independent of all the other sets and consists of a specific collection of related database-tree structures. Each entry-point set can have its own unique tree structure, and therefore each can represent a given scenario with an independent database design. Separate linked databases could be used in the same manner as entry-point sets within a database. A scenario-dependent database provides great flexibility. Since each entry-point set (or separate database) has its own tree structure, each can have its own attendant dynamic and static models.

4.3 Model Selects

The databases were organized so that the viewpoint processor makes a series of hierarchically structured choices, or model selects, that determine the requirements for any given scenario. A CT6 image generator allows 128 different model selects for each dynamic coordinate system and has the unique capability of performing model selects of dynamic coordinate systems in real time. This feature is often used for animation. Because the top-level structure is designed so that there are several mutually exclusive model selects, when the viewpoint processor starts its trace of the database tree it also starts to determine the current scenario's requirements based on the model selects chosen for that field by the simulator-host-computer program. Each of the top-level model selects points to another structure that in turn is attached to a dynamic coordinate system with multiple model selects. The viewpoint processor continues this tracing of nested model selects until it incorporates all the specified scenario requirements.

5.0 DESIGN CONSIDERATIONS

Once the organization of a space database is determined, other aspects of the design can be considered. Some of the main issues and challenges encountered in the design of the space databases, the resolutions of the challenges, and the ramifications of the decisions made concern the database's coordinate system, database units, the construction and placement of models, the uses of texture, collision detection, the simulation of closed-circuit-television systems, and the overall modularity and flexibility of the databases.

5.1 Coordinate System

NASA's space databases were designed with an orthogonal Cartesian coordinate system. If the simulator-host-computer program were to use a different coordinate system than the image generator does (a celestial coordinate system, for example), all motions controlled by the simulator host computer would have to be transformed to map that coordinate system into the image generator's coordinate system.

All of the NASA models were constructed about or relative to the origin of the image generator's coordinate system. However, the placement of the origin, the database unit of measure, and the overall size of the database affected the design of the individual models.

The most likely location for the database's global origin is the center of geometry (or center of mass) of the main scenario model. The Orbiter's origin was chosen as the global origin of the database that included the Orbiter, SRMS, and Space Telescope models. Each of these models was easily defined relative to the origin of the Orbiter, and every scenario maintained the Orbiter as a static model at the global origin. However, this origin was not appropriate for the other two NASA databases, and therefore other origins were selected.

5.2 Database Units

The size or volume of a database is limited by the highest number the image generator allows and by the chosen unit of measure. A CT6 image generator does not use an inherent unit of measure as a database unit; it simply interprets all database values as numbers with a maximum and minimum range. Any unit of measure can be used to create a database as long as the relative scale between database models is maintained. The most common database unit for terrain databases is the foot. The database unit chosen for the space databases was the inch.

5.3 Models

The models used in a space simulator must be extremely accurate. They are the only visual cues in the database because black space, unlike terrain, does not provide much in the way of feedback. The models are also generally viewed at a much closer range than models in a terrain database, so they need to include as much of the geometry of the real-world object they represent as the image generator can process. Most scenarios require extreme accuracy in motion, collision detection, and interaction, so the models must provide all of the cues needed to make a given scenario effective.

Each model in the space databases was constructed with information derived mostly from detailed engineering drawings, but because many of the actual objects had not yet been constructed, the models were continually subject to reevaluation. The Orbiter was the only item that had been constructed before development of the databases began. Because such high fidelity was required of the models, the Orbiter model geometry was created through a process that involved automatically scanning a scaled replica, digitizing hundreds of vertices, and defining the polygonal boundaries. Although the process was involved, it was very efficient and quickly accomplished, and the end result was extremely accurate.

Since the chosen database unit and the maximum value allowed by the image generator limited the overall size of the database, some of the database models could not be constructed at a 1:1 scale. Instead, they had to be created at other scales and then positioned to appear their correct sizes and to maintain the integrity of the simulation.

One of the requirements specified for the space databases was an accurate starfield. The inclusion of stars of various intensities and locations in a sea of black greatly enhances the illusion of space. Space without stars is about as useful as unembellished terrain is for a flight simulator. Digistar, a star-projection system developed at Evans & Sutherland for planetariums, provided the information used to create the starfield for the space databases.

The starfield model has more than 1600 unique stars that appear to be in their correct locations. The model is a sphere consisting of light points; the center of the sphere is at the global origin of the database. The sphere was scaled to fit within the allowable database size and each light point was assigned an intensity that would simulate the apparent magnitude of the star it modeled. Because many constellations can be recognized in this starfield model, it could easily be used for directional information.

Two static models and dynamic-texture motion were used to create the illusion of an orbiting satellite and a rotating Earth. The database unit coupled with the maximum size of the database determined that a full-sized Earth could not be modeled. Therefore, a scaled model of Earth was created and placed artificially close to the orbiting vehicles. This strategy effectively created the illusion of an accurate orbit and met the requirements of the application.

To complete the illusion of Earth as seen from an orbiting satellite, photo-derived dynamic texture was applied to the Earth model and given a directional velocity that approximates the speed at which a satellite such as the Orbiter passes over a given point on Earth's surface. So the simulated orbit is in essence the opposite of what happens in the real world: the orbital vehicle is static and Earth moves by below it. This effect was possible because the CT6 image generator has the ability to create real-time texture motion.

Because Earth, in this case, could be built as a static model, it was modeled as a disk rather than as a sphere. Modeling Earth as a disk makes it difficult to simulate its dark side, but this disadvantage is offset by the fact that valuable dynamic coordinate systems and viewpoint-processor time are preserved for other uses. However, the same illusionary techniques could be used to construct a spherical model of Earth.

Since there is no atmosphere to interfere with light in space, colors are more vibrant than on Earth and there is a crisp contrast between light and dark. Therefore, color assignments, usage, and tuning are important design factors. Since the visual-system displays cannot reproduce the contrasts found in space exactly, relative brightnesses must be assigned to certain models. No model should be colored as black as the blackness of space, for example.

5.4 Texture

2D texture can realistically enhance computer-generated imagery and has the great advantage of generally not affecting the processing load. Because texture in a CT6 image generator is processed in parallel with polygons, a scene consisting of the maximum number of processable polygons can be enhanced by the addition of texture and still not overload the image generator.

Since the fidelity of the models was an important criterion of the space-database requirements, texture was often used to effect visual cues without increasing the number of polygons. These applications of texture significantly enhanced the Earth and Orbiter models.

The entire surface of Earth is simulated by a photo-derived, self-repeating texture map applied to a group of polygons. The Earth model, because of its special texture application, maintains a slim polygon budget and is therefore a very efficient model.

The Orbiter was also modeled with texture. The insignia on the outside surfaces were created with texture. The American flag, all thirteen red and white stripes and all 50 stars on a blue background, was created with one texture map and four polygons. Photo-derived, self-repeating patterns adorn the inner bay with a thermal wrap pattern and the outer skin with tiles.

All of the additional visual cues created with texture allow the image generator's polygon capacity to be spent where texture cannot be used, as for the hundreds of truss polygons on the Space Station.

Simple 2D polygons with texture applied to them can simulate complex 3D structures. The most notable of these illusions is the texture application that simulates the inside volume of the End Effector (EE) with just one polygon. The EE is essentially a hollow cylinder whose inside portion is seen from many eyepoints. Since the use of polygons on an inside surface was considered wasteful, a special texture pattern was applied in a plane other than that of the polygon that capped the end of the EE. The resulting parallax created an illusion of depth that is almost indistinguishable from that of an actual modeled volume. Noncoplanar texture is another robust capability of the CT6 image generator without which this illusion could not have been created.

5.5 Collision Detection

One of the most important functions of a space simulator is the detection of collisions between various objects. The collision-detection function can be used for purposes other than detecting when two models collide, however. For example, the complex simulator task of grappling a payload with the SRMS would be virtually impossible without the aid of collision detection. An appropriate viewpoint-processor-timing budget must include enough time for all of the collision-detection requirements.

Collision detection is another reason for constructing accurate models. The database must reproduce the real world in order for the simulation to be effective. Otherwise, collision detection might not be reliable.

5.6 Closed-circuit Television

In an actual space mission, a significant portion of the scenes outside the Orbiter are viewed through closed-circuit-television (CCTV) systems. These sophisticated systems have a built-in process called automatic level control (ALC) that adjusts the CCTV according to the light sources it receives. Remotely similar to the light meter and *f*-stop combination on a camera, ALC is an automatic process used to improve and adjust image quality.

One of the requirements of the space databases was to simulate the effects of ALC. The simulation is done through detection of light from its various sources (sun, moon, Earth, and floodlights) and concomitant adjustment of the scene illumination. Unique structures had to be created on each of the light-source models. The image-generator function used these unique structures to detect when the light-source models came into view. The real-time system would notify the simulator-host-computer program when the models came into view, and the simulator-host-computer program would then adjust the illumination of that channel. Without the characteristics of the CCTV system and ALC, every scene would appear as though it were an out-the-window view and the simulation would not be accurate.

5.7 Modularity and Flexibility

The structure of a space database must be modular and flexible because the database will probably be used for many applications over a wide range of projects. Therefore, each model must be designed, organized, and documented so that another engineer can make modifications to it in the future. Furthermore, the top-level structure must be clean, concise, and simple, with adequate and descriptive documentation. This is especially important for scenario-dependent databases or fully compatible model-select databases. The criteria of modularity and flexibility are arguments for simple databases over extremely large, complex, all-encompassing databases.

6.0 CONCLUSION

The design and development of a space database is a very challenging and rewarding experience. Although all real-time visual databases are hierarchically-ordered groups of geometric data, the differences between familiar terrain databases and space databases are significant. The organization of a space database is substantially dependent upon the simulation and should meet the requirements of specified scenarios. Hence, the number of and the relationships among the static and dynamic models are crucial to a successful design. More dynamic models and better image-generator performance result from a well-organized database design. The scenario requirements and all of the database requirements must be taken into account when the visual system's resources are budgeted so that the most processing-efficient and visually-effective database possible is designed.

Furthermore, the potential for improved scene realism is generally greater for a space database because of the ability of computer-generated imagery as a medium to depict regular geometric items and because of the visual quality of photo-derived-texture applications. Highly efficient and accurate models are essential not only for visual cueing but for supporting other visual-system functions such as collision detection. Finally, well-engineered database source code and its accompanying documentation provide service and support for many future project applications.

7.0 ACKNOWLEDGEMENTS

A considerable amount of research was involved in creating the NASA space databases. The entire development team, consisting of NASA and Evans & Sutherland personnel, participated in these innovative designs and thereby contributed greatly to the content of this paper. Specifically, Mike Bartholomew, David C. Christianson (NASA), Mercedes DeLugo, Ralph Howes, Michael D. Jackson, Daniel Lunt, Janice Poulson, and James R. Smith (NASA) deserve special recognition for the information, guidance, and assistance they provided. Curtis G. Booth rendered invaluable technical and professional services in the preparation and organization of this document.