A METHOD AND SYSTEM FOR COMPRESSING AND DECOMPRESSING DATA

Technical Field

The present invention relates to a method and system for compressing and decompressing data, and in particular, for compressing and decompressing data contained in scene files representing scenes to be displayed in a computer animation.

Background of the Invention

In the area of computer animation, "scene files" contain data which represents a particular scene or sequence of frames in a computerised animation. Two commonly used scene file formats include the *RenderMan*® and *mental images*® scene file formats. For instance, *RenderMan*® scene files are written in a scene description language which describes such things as objects, scenes, lights, and cameras so that a computer program can render photo-realistic images from them.

Typically, scene files are exported from three-dimensional modeling applications before being transported across a communication network for processing by rendering clusters. Unfortunately, scene files can each be several gigabytes of data in size so as to make it infeasible to transfer such large scene data files across the communication network due to bandwidth and time constraints or even to submit on different media (DVD, tapes). The large size of scene data files is at least to some extent due to the duplication of "object data" in the scene data files – that is, data representing an object (typically a three-dimensional object) that is to be rendered on a display from the scene data file.

One approach to overcoming this problem has been to reduce the complexity of each scene in order to achieve a smaller file size which in turn speeds up the transfer time of submitting the scene files. However, in many cases, this approach is undesirable as it compromises the overall quality of the computer animation.

Prior art systems such as that used by *mental images*® have employed three-dimensional modeling software that converts all animation frames into a single scene file. Whilst some degree of compression is achieved in carrying out this process, it is inconvenient in that it involves a relatively high degree of manually intensive labour – that is, animators are required to manually

identify associated files containing texture and shading information and thereafter use commercial compression software to package both the texture and shader files together with the actual compressed scene data file into a single compressed file for transport to a remote rendering cluster.

Another problem with prior art systems such as that used by *mental images*® is that in order to package the animation frames into a single scene file for transport, the frames must be in running sequence. This is not always practical in that, in some cases, where animators are required to re-render many non-sequential blocks with small frame sequences, these blocks are usually converted separately such that the optimal compression ratio cannot be achieved.

Attempts have been made to compress *RenderMan*® type scene files using what is referred to as a *diff* method which involves the use of a universal "Lowest Common subsequence algorithm" to compare each and every frame in a frame sequence with the first frame in order to find a common subsequence – that is if any data representing a scene object in a scanned frame is not contained in the first frame of the sequence, then the object will not be compressed even if the object is contained in several other frames in the sequence. The *diff* method is a relatively time-intensive process particularly where it is processing data of several gigabytes in size. Another problem with *diff* is that is will not compress several separate frames into a single file and nor does it have an awareness of the actual content of the scene file.

Summary of the Invention

The present invention seeks to alleviate at least one of the problem discussed above in relation to the prior art.

The present invention may involve several different broad forms. Embodiments of the invention may include one or any combination of the different broad forms herein described.

In a first broad form, the present invention provides a method for use in compressing a scene data to form a compressed scene data, the scene data being representative of a scene to be rendered on a display of a computing system and the scene data including a plurality of object datas which are each representative of an object in the scene to be rendered on the display, wherein the method includes the steps of:

(i) identifying a first set of object datas in the plurality of object datas, wherein the first set of object datas are representative of substantially a same object to be rendered on the display;

(ii) substituting each of the object datas in the first set of object datas in the scene data with an associated hash code of a first object data in the first set of object datas so as to generate a compressed scene data from the scene data; and

whereby, the compressed scene data is able to be decompressed to produce the scene data by substituting each of the hash codes in the compressed scene data with the associated first object data.

In a second broad form, the present invention provides a method for use in decompressing a compressed scene data produced in accordance with the first broad form of the present invention, the method including the steps of:

(i) identifying at least one hash code in the compressed scene data;

(ii) determining the first object data associated with the at least one hash code;

(iii) substituting the at least one hash code in the compressed scene data with its associated first object data such that scene data is generated from the compressed scene data.

Typically the object is a three-dimensional geometric object to be rendered on the computerised display from the scene data.

Preferably, the hash code associated with the first object data may be of relatively smaller size than its associated first object data. Advantageously, this may enable the size of the compressed scene data to be relatively smaller than the size of the scene data for storage and/or transmission via a communication channel. Typically, the hash code associated with the first object data may be generated by using a "Message-Digest Algorithm 5" type hashing function to process the first object data.

Preferably, the hash code associated with each object data in the first set of object datas is the same and thus it is usually the case that only one object data in the first set of object datas is processed by the hashing function to obtain a hash code which may be substituted into the scene data for all of the object datas in the first set of object datas.

Typically, there may be multiple sets of common object datas within a scene data wherein each set of object datas represent a uniquely different object to be rendered on the display. Typically, each set of object datas may have a unique hash code associated with it and during compression. The unique hash code may be substituted for all object datas in the scene data which contained in a set of object datas associated with that hash code.

Preferably, the present invention may include the step of recording into a dictionary, the hash code associated with the first set of object datas. Typically, the dictionary may also provide a link between the hash code and its associated first object data such that the first object data may be identified during decompression of the compressed scene data. Advantageously, when the compressed scene data is being decompressed, whenever a hash code is detected in the compressed screen data, the dictionary may be queried in order to provide a reference to its associated first object data which should be substituted in place of all occurrences of the associated hash code in the compressed scene data.

Typically, the scene data representing a plurality of frames which form a sequence in a computerised animation.

Typically, the scene data may be sourced from at least one of a *RenderMan*® and/or *mental images*® type scene file.

Typically, the scene data is compressed in a lossless manner.

Preferably, the present invention also includes the step of automatically extracting texture/shader files from physical storage that associated with the scene data wherein the texture/shader files may represent a particular texture/shading characteristic of an object to be rendered on the display. Advantageously, the present invention automates the extraction of texture/shader files which, in contrast to certain prior art techniques, typically requires an animator to manually extract texture/shader files from physical storage that associated with the scene data in which case, a missing texture/shader file might cause the scene to be incorrectly rendered. Consequently, this will affect the budget and the delivery schedule of the animators.

Preferably, at least one of the compressed scene data containing the dictionary and the extracted texture/shader files may be bundled into a bundled file for storage and/or transport across a

communication network to, for instance, a rendering cluster for further processing. Advantageously, the present invention assists in processing the scene data as a single data block – that is, where the scene data represents a multitude of frames for an animation, all objects contained in the frames of the scene data may be processed collectively and this contributes towards an improved compression ratio.

Typically, the present invention may also include the step of further compressing the bundled file. For instance, the bundled file may be further compressed using compression software such as *gzip* and the like.

In a third broad form, the present invention provides a computerised system adapted for performing the method steps in accordance with at least one of the first and/or second broad forms of the present invention.

Typically, the computerised system may include a processor and a memory store operatively connected to the processor, wherein the memory store contains a program executable by the processor to automate the steps of at least one of the first and/or second broad forms of the present invention.

Typically, the computerised system may include a distributed network including an origin communicably connected to a destination via a communication network or any other transport media. Typically, the communication network may include at least one of a LAN, a WAN, an intranet or the Internet.

Typically, the origin may include a compression module programmed to perform the method steps in accordance with the first broad form of the present invention. Also typically, the destination may include a decompression module which is programmed to perform the method steps in accordance with the second broad form of the present invention.

In a fourth broad form, the present invention provides a storage device adapted for storing a computer program adapted for performing the method steps in accordance with at lest one of the first and/or second broad forms of the present invention.

Typically, when the scene data is scanned for object data and/or texture/shader files, the present invention may scan the scene data at the content level rather than the binary level. For instance, this may involve scanning the scene data for markers in the file which may indicate the leading and ending points of an object data and/or texture/shader files. Advantageously, the present invention may be aware of the structure of the scene data so as to know where to start looking for object data and/or texture/shader files, thus resulting in reduced compression and decompression times. In contrast, prior art systems employ generic search algorithms which generally are not content-sensitive, and therefore tend to take longer to process the scene data.

Brief Description of the Drawings

The present invention will become more fully understood from the following detailed description of a preferred but non-limiting embodiment thereof, described in connection with the accompanying drawings, wherein:

Figure 1 depicts a computerised system which is used to execute a first embodiment method of compressing and decompressing scene data.

Figure 2 depicts a flow diagram depicting method steps for compressing a scene file in accordance with the first embodiment which has been compressed at the origin;

Figure 3 depicts a flow diagram depicting method steps for decompressing a compressed scene file in accordance with the first embodiment which has been transmitted to the destination from the origin.

Modes for Carrying Out the Invention

A first embodiment of the present invention involving a method of compressing and decompressing scene data during transmission of the scene data across communication channel will now be described as follows with reference to Figs. 1 to 3.

The method includes connecting the origin (100) and destination (200) via the Internet/network (300) or any other transport media. The origin (100) includes a compression module consisting of a compression software application which is adapted to compress the scene data in accordance with the first embodiment method whilst the destination (200) includes a decompression module consisting of another software application adapted to decompress the compressed scene data in accordance with the first embodiment method. In this embodiment, by way of example, the destination includes a rendering server.

The scene data described herein relates to data contained in a scene file formatted in accordance to RenderMan® standard.

Referring now to Fig. 2, in use, the compression module residing in the origin (100) begins by scanning the scene data contained in the *RenderMan®* scene file in order to identify if there is any object data. This step is broadly represented at block (1) in Fig, 2. As the contents of the scene data are being progressively scanned for object data and/or texture/shader files, the contents are appended to a log file either without modification (ie, when an object data and/or texture/shader files are not encountered) or with modification (ie, when an object data and/or texture/shader files are encountered.

As soon as the compression module encounters an object data, it computes a hash code for the object data as shown at block (2) in Fig. 2. In this embodiment, a "Message-Digest Algorithm 5" type hashing function is used.

The compression module detects object data by scanning for starting and ending points of object data by reference to known *RenderMan® standard.* For instance, Table 1 shows the leading and ending text markers defining an object data in the *RenderMan®* and *mental images®* file formats.

	Lead mark	**End mark**
RenderMan®	PointsGeneralPolygons	"/n" (newline)
mental images®	"group"	"end group"

Thereafter, the compression module compares the computed hash code of the object data with existing hash codes contained in a dictionary. If the dictionary does not currently include an entry matching the calculated hash code, then a new entry is entered into the dictionary with the hash code as well as a link to the actual object data associated with it; The steps are broadly represented at block (4) in Fig. 2.

In the event that the dictionary is found to contain a matching reference to the calculated hash code, then the computed hash code is simply written into the log file in substitute for the encountered object data. This step is broadly represented at block (5) in Fig. 2.

The process of scanning for object code is repeated until all object code has been detected and processed accordingly. Thereafter, the compression module automatically scans the scene data

again in order to detect and extract texture/shader files. Whenever texture/shader file is encountered, it is extracted and bundled with the log file that contains the dictionary. This step is broadly represented by the blocks labeled (6) in Fig. 3.

When all texture/shader files have been extracted and with the bundled log file that contains the dictionary, they undergo further compression by a compression software application, before being transported to the destination via the Internet/network or any other transport media where it will undergo decompression.

Figure 3 shows a flowchart outlining the broad steps involved in decompressing the compressed scene data. At destination (200) end, a software application is adapted to decompress the compressed scene data. This begins with the application firstly unzipping the received file to remove the first layer of compression.

Thereafter, destination (200) unpacks the texture/shader files onto a physical location. This step is broadly represented by block (7) in Fig. 3.

Thereafter, the log file is scanned by the decompression module in order to detect any hash codes. Whenever it identifies a hash code, the application refers to the dictionary to identify the associated object data that should be retrieved from the dictionary by reference to the detected hash code. The retrieved object data is substituted into the compressed scene file in place of the hash code. This is repeated until all hash codes in the compressed file have been substituted with their associated object data. The original uncompressed scene file is thereafter produced again once all substitution has been completed. This step is broadly represented by the blocks labeled (8) in Fig. 3.

The reference in this specification to any prior publication (or information derived from it), or to any matter which is known, is not, and should not be taken as an acknowledgment or admission or any form of suggestion that that prior publication (or information derived from it) or known matter forms part of the common general knowledge in the field of endeavour to which this specification relates.

Those skilled in the art will appreciate that the invention described herein is susceptible to variations and modifications other than those specifically described without departing from the scope of the invention. All such variations and modification which become apparent to persons

skilled in the art, should be considered to fall within the spirit and scope of the invention as broadly hereinbefore described. It is to be understood that the invention includes all such variations and modifications. The invention also includes all of the steps and features, referred or indicated in the specification, individually or collectively, and any and all combinations of any two or more of said steps or features.

WHAT IS CLAIMED IS:

1. A method for use in compressing a scene data to form a compressed scene data, the scene data being representative of a scene to be rendered on a display of a computing system and the scene data including a plurality of object datas which are each representative of an object in the scene to be rendered on the display, wherein the method includes the steps of:

(i) identifying a first set of object datas in the plurality of object datas, wherein the first set of object datas are representative of substantially a same object to be rendered on the display;

(ii) substituting each of the object datas in the first set of object datas in the scene data with an associated hash code of a first object data in the first set of object datas so as to generate a compressed scene data from the scene data; and

whereby, the compressed scene data is able to be decompressed to produce the scene data by substituting each of the hash codes in the compressed scene data with the associated first object data.

2. A method for use in decompressing a compressed scene data produced in accordance with claim 1, the method including the steps of:

(i) identifying at least one hash code in the compressed scene data;

(ii) determining the first object data associated with the at least one hash code;

(iii) substituting the at least one hash code in the compressed scene data with its associated first object data such that scene data is generated from the compressed scene data.

3. A method as claimed in claim 1 or 2 wherein the object includes a three-dimensional geometric object to be rendered on the computerised display from the scene data.

4. A method as claimed in any one of the preceding claims wherein the hash code associated with the first object data is of relatively smaller size than its associated first object data.

5. A method as claimed in any one of the preceding claims including the step of generating the hash code associated with the first object data using a "Message-Digest Algorithm 5" (MD5) type hashing function.

6. A method as claimed in any one of the preceding claims wherein the hash code associated with each object data in the first set of object datas is the same.

7. A method as claimed in any one of the preceding claims including the step of recording hash codes and their associated object datas into a dictionary.

8. A method as claimed in any one of the preceding claims wherein the scene data includes data contained in at least one of a *RenderMan*® and/or *mental images*® type scene file.

9. A method as claimed in any one of the preceding claims including the step of automatically extracting texture/shader files from a physical location that associate with the scene data.

10. A method as claimed in any one of the preceding claims including the step of further compressing the log file and the associate texture/shader files of the scene with compression software.

11. A computerised system including a processor and a memory store operatively connected to the processor, wherein the memory store contains a program executable by the processor to automate the method steps in accordance with any one of claims 1 to 10.

12. A computerised system as claimed in claim 11, wherein the origin is operatively connected to the destination via a communication network.

13. A computerised system as claimed in claims 11 or 12, wherein the origin includes an executable compression module for compressing the scene data, and the destination includes an executable decompression module for decompressing the compressed scene data in accordance with the method steps of any one of claims 1 to 10.