Planning/Scheduling Techniques for VQ-Based Image Compression

Nicholas M. Short, Jr.
Code 935, Goddard Space
Flight Center, Greenbelt, MD 20771

Mareboyana Manohar
Hughes STX, Code 935
Goddard Space Flight Center
Greenbelt, MD 20771

James C. Tilton
Code 935, Goddard Space
Flight Center, Greenbelt, MD 20771

Abstract

The enormous size of the data holdings and the complexity of the information system resulting from the EOS system pose several challenges to computer scientists, one of which is data archival and dissemination. More than ninety percent of the data holdings of NASA is in the form of images which will be accessed by users across the computer networks. Accessing the image data in its full resolution creates data traffic problems. Image browsing using a lossy compression reduces this data traffic, as well as storage by factor of 30-40. Of the several image compression techniques, VQ is most appropriate for this application since the decompression of the VQ compressed images is a table lookup process which makes minimal additional demands on the user's computational resources. Lossy compression of image data needs expert level knowledge in general and is not straightforward to use. This is especially true in the case of VQ. It involves the selection of appropriate codebooks for a given data set and vector dimensions for each compression ratio, etc. A planning and scheduling system is described for using the VQ compression technique in the data access and ingest of raw satellite data.

1 Introduction

Over the next decade, the rate at which data is generated by space-borne instruments will increase dramatically over current levels. A major contributor to this increase is the Earth

Observation System (EOS), planned for the end of this decade. The five proposed instruments on the EOS AM-1 platform and the six proposed instruments on the EOS PM-1 platform will generate data at a combined rate of 281 Gigabytes per day.

This raw data generated by the EOS platforms will be in turn processed into data products, including radiometrically and geometrically corrected images and a large number of science data products. This increases the data volume that must be handled and stored from the EOS instruments by an order of magnitude. Thus, over one Terabyte of EOS data products will be stored each day, along with other Earth science data, in distributed active archive centers (DAACs) located throughout the United States. Over the 15-year life of EOS, the archives will manage 11 petabytes of raw, processed, and analyzed data.

Success of future Earth science missions depends upon increasing the availability of data to the scientific community who will be interpreting space-based observations for issues such as ozone depletion and greenhouse effects, land vegetation and ocean productivity, and desert/vegetation patterns to name a few. Part of NASA's role in the Mission to Planet Earth (MTPE) initiative is to take a proactive leadership role in the management of space and Earth science data and in making those data accessible to scientists worldwide in order to foster the new field of Earth Systems Science.

Even at current data volumes, it is difficult to design and operate effective data archive and distribution systems for NASA Earth science data archives. With the increasing volumes of data that will be stored in these data archives, efficient browsing and distribution of data from these archives becomes even more important. An effective data archive and distribution system must give quick access to image browse and other data so users may quickly select the data required for their application. The availability of image data at intermediate resolution levels would also help users resolve ambiguities in the data selection process.

From our research in the Information Science and Technology Branch (ISTB), we present here an image browsing scheme using VQ and progressive VQ compression algorithms that we claim are excellent candidates for image data browsing and retrieval. A key feature of VQ and progressive VQ is their asymmetry in encoding and decoding. The minimal computational requirements of progressive VQ for decoding make possible very quick retrieval on moderate computer systems. The more computationally intensive encoding process can be accomplished, at a sufficient rate to keep up with the incoming data flow, in centralized data processing centers using more powerful computers, such as the recent massively parallel models.

To compress image data an expert level of knowledge is required. For example, a VQ or progressive VQ based image compression needs information about the data and the instrument the data belongs to, vector dimensions, etc. for selecting the codebook for compression. Usually the user has no knowledge of this information. However, the user is primarily concerned about the compression ratio and quality of the compressed image. Therefore, a planning/scheduling system is required that accepts the user specified parameters and translates them to VQ related parameters. Thus the Planner/Scheduler essentially helps eliminate an image compression specialist from data dissemination process.

2 Image Compression

Image compression is one of many tools that can be used to help address Mission to Planet Earth's data handling challenges [23]. However, no single data compression approach is likely to be appropriate for all aspects of the problem. Lossless compression is required for data archiving, while some degree of information loss may be allowable for video image transmission. For image browse applications, larger amounts of information loss may actually be desirable. For browse, a general overall impression of the data quality and content may be all that is necessary, and a large reduction of data volume may be required. The key task for lossy data compression for browse applications is to preserve only the information required. Data characteristics also must be considered in designing an appropriate data compression approach, since data compression approaches often assume a particular data model.

Earth scientists often need to browse data to check the appropriateness and quality of particular data sets for detailed analysis. Further, appropriately derived browse data can facilitate interdisciplinary surveys which search for evidence of unusual events in several data sets from one or more sensors. In addition, browse data can be used to validate the quality of the data by facilitating quick checks for data anomalies. These different uses of browse data put possibly conflicting requirements on the browse data, and may require that separate browse data sets be produced for each major use category.

If a "progressive" data compression approach [23] is used, browse data can also facilitate the distribution of the data from the archive. Here the image is compressed at various levels called a compression hierarchy. The first level of the hierarchy provides an initial rendition appropriate for browsing the data. The ensuing levels of the hierarchy contain the details that are missing at earlier levels. Either a user or the planner/scheduler would inspect the browse data, and decide at "anytime" whether or not to inspect the data more closely. If a closer inspection is desired, additional levels of the compression hierarchies would be requested, until the user decides that data is not appropriate for the application and terminates accessing the data set, or until fully reconstructed data is obtained. Under this scheme, the data distribution process is kept efficient since no redundant information is ever sent or used.

Many image compression approaches show promise for the data archive and distribution problem. These include the Joint Photographic Experts Group (JPEG) standard lossless and lossy compression methods [18], the Rice algorithm [19, 20], variations on Vector Quantization [10, 1, 14]. In addition, combinations of subband/wavelet decomposition and Vector Quantization [2, 3, 11], and combinations of subband/wavelet decomposition with the Karhunen-Loeve transform [8, 15] also show promise.

We have concentrated our efforts on investigating image compression based Vector Quantization. These approaches are particularly suitable for data archives and distribution across computer network applications due to asymmetrical coding and decoding efficiencies. The coding is computationally expensive, but is a one time effort, and can be performed at an archival center using a large capacity machine. The decoding part, however, is a computationally inexpensive table lookup process which does not burden the end user with computational difficulties.

2.1 VQ and Progressive VQ

VQ is the vector extension of scalar quantization which is found to be very useful for multispectral image compression ([13, 15]). The VQ vectors are obtained from image data by systematically extracting nonoverlapping blocks (typically 4x4) and arranging the pixels in each block in raster scan order. Such vectors allow VQ to exploit two dimensional correlations in the image data. If the image is multispectral, nonoverlapping cubes (typically 4x2x3) may be used. VQ builds up a dictionary of a few representative vectors, called codevectors, and then codes the image with the index value of the closest codevector from the dictionary, called codebook, in place of of each vector. Each codevector is represented by an address containing log_2M bits, where M is number of codevectors in the codebook. Assume vectors of size k are drawn from the input image and matched with those in the codebook. Using the indices of the matched codevectors to represent the input image vectors results in a decreased rate of $(log_2M)/k$ bits/pixel or a compression ration of $(k * n)/log_2M$, where n is the radiometric resolution of the image. In all practical situations the codebook size, M, is much smaller than the number of vectors that make up the input image.

The most important phase of VQ is the training process in which an optimal codebook (by some criterion such as least MSE) is learned from the input samples. The most widely used algorithm is Linde-Buzo-Gray (LBG) algorithm ([10]). Both the training and coding phases of VQ require finding the codevector which is closest match to a given vector. Computing this closest match requires computations proportional to the size of the codebook. Computational cost can be reduced by employing a suboptimal approaches such as Tree Search Vector Quantization (TVSVQ) and Pruned Tree VQ (PTVQ) [10]. The computational problems can also be solved by using a special architectures [13].

Progressive VQ [14] is a progressive variant of VQ in which multiple compression levels are provided. The first level is a VQ coding in which the codebook and codevector parameters are adjusted to give a relatively high compression ratio (e. g., in the range of 30 to 50). The image reconstructed from this first level coding can serve as browse data for a data archive system. If n levels are used, the second through the n-1 levels are VQ coded residuals. The nth level residual is not VQ coded, but instead is encoded with a lossless approach, such as the Rice algorithm [20] or Ziv-Lempel algorithm [25].

3 Planning and Scheduling for Image Compression

Given that image compression, like many other image processing routines, has many possible variants and uses, selection and coordination of the appropriate routines for particular users needs requires the use of a supervisor function. Many researchers have suggested the application of rule-based expert systems for capturing user requirements and knowledge for image processing[16, 17, 6, 21]. However, none of these techniques explicitly takes into account the computational complexity or the resource requirements for image processing tasks. In this domain where computational resources are constrained and hard deadlines for data acquisition exist, a better model that combines knowledge representation with resource modeling needs to be incorporated.

Recently, researchers have suggested the use of AI planning /scheduling techniques to manage the coordination of image processing operators such as image compression[7, 22, 5, 12]. For this paper, we will illustrate a particular planning /scheduling approach, called PlaSTiC, which is being used at the ISTB.

PlaSTiC was developed by the ISTB and Honeywell Technology Center as a planning /scheduling tool for a distributed computing environment. PlaSTiC is a hierarchical planner loosely based upon work by [24]. The core system is based upon the Honeywell's Time Map Manager (TMM) that handles reasoning about temporal information[4]. PlaSTiC combines the Nonlin planner[9], TMM, and extensions that allow for reasoning about the duration and resource requirements of plans[5].

For the image processing, plans are handed to an execution monitor which interprets plans according to the run-time environment, assigns uncommitted tasks to processes, and collects statistics for the planner. These statistics provide best-case/worst-case estimate intervals for primitive tasks and are propagated back up a task formalism [5] to provide better constraints during task decomposition.

As with most planners, PlaSTiC maintains a knowledge-base of plan operators that during planning, provides the necessary knowledge for plan construction. As an example, contrived plan operator for PVQ compression, consider the following:

```
(opschema pvq-compression :todo (file-format ?FileID PVQ-COMPRESSED)
                          :expansion ((step1 :goal (file-format ?FileID BINARY))
                                      (step2 :goal (file-format ?FileID BSQ))
                                      (step3 :primitive (PVQ-COMPRESS ?name ?name ?cname
                                                          ?c ?r ?x ?y ?n IDM))
                                      (step4 :primitive (UNIX-COMPRESS ?name ?cname UNIX)))
                          :orderings ((before step1 step2) (before step2 step3) (before step3 step4))
                          :conditions ((:use-when (name ?FileID ?name))
                          (:use-when (size ?FileID (?r ?c)))
                          (:use-when (codebook ?FileID ?codebook))
                          (:use-when (codebook-name ?codebook ?cname))
                          (:use-when (vectx ?codebook ?x))
                          (:use-when (vecty ?codebook ?y))
                          (:use-when (codebook-band-number ?codebook ?n)))
:duration                 (range-addition (file-format-estimator 2)
                                          (pvq-estimator ?n ?r ?c ?x ?y))
:variables (?x ?y ?n ?r ?c ?FileID ?name ?cname ?codebook))
```

Essentially, the above pvq-compression operator states that in order to put a file (represented by the variable ?FileID) into PVQ compressed format (i.e., via the todo slot), two goals (i.e., step1 and step2) for putting the file in binary and binary sequential format must be done before the pvq-compress command (step3) gets called. In this case, each of the steps are totally ordered[1]according to the orderings slot. This operator is only applicable if there exists the appropriate information specified by the conditions slot.

In PlaSTiC, information about the duration of these operators is specified either explicitly through the duration slot above or through a statistical gathering mechanism that sets the duration of primitive steps (e.g., steps 3 and 4 above). Durational information is specified as a range of values from a lower bound to an upper bound. For operators with the duration slot, a function can be specified that must return a range. This function's arguments are derived through variables that are bound from the conditions slot[2].

Typically, the function in the duration slot is either a statistical estimator or a polynomial (e.g., *big oh* notation). Examples of the former can be as simple as returning the min/max of a working set or as complicated as output from an unsupervised clustering where attributes can be any property from the execution environment such as CPU utilization, machine type, input size, etc. For the primitive steps, durations are only min/max values from a working set.

[1] (before step1 step2) means step1 occurs before step2

[2] Actually, unbound variables can exist as well, but that requires a more complicated mechanism

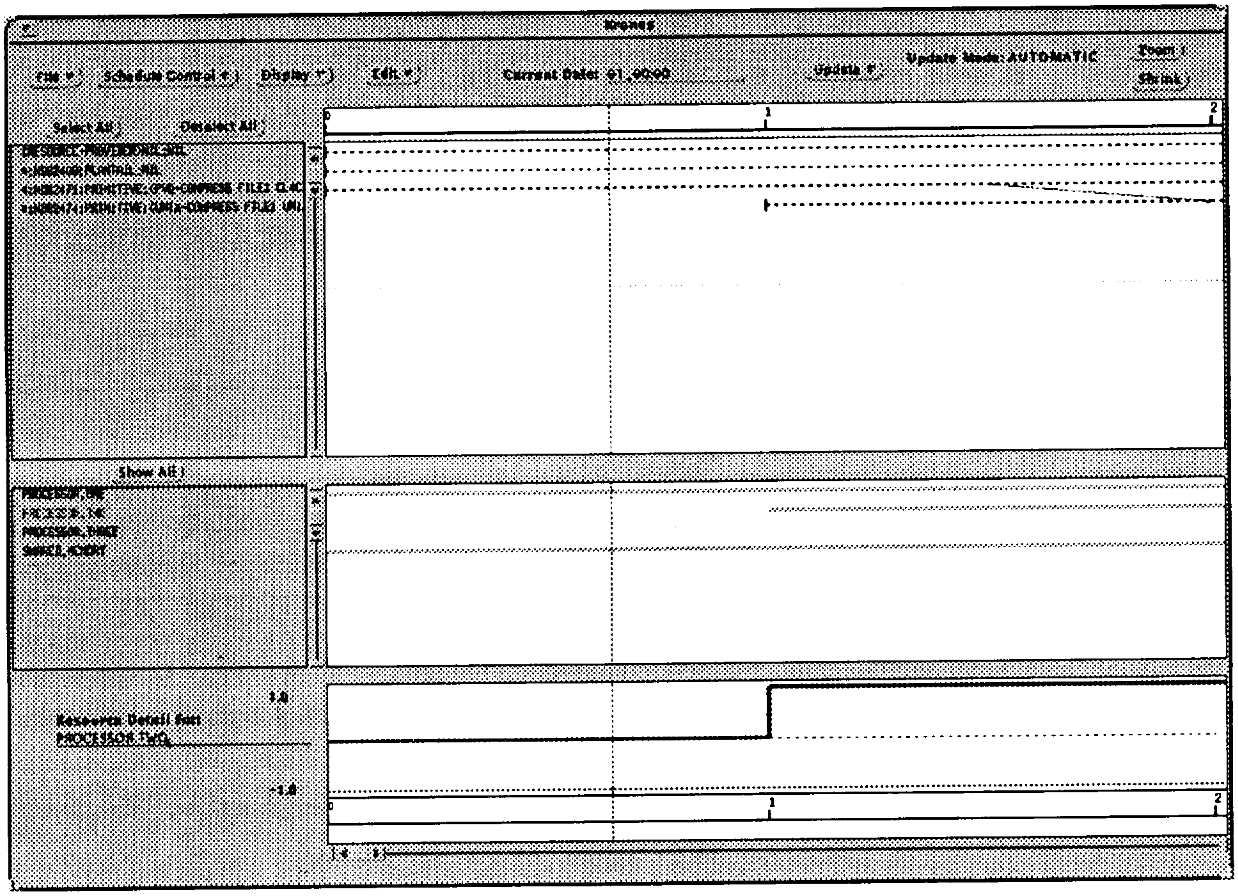

Figure 1: Plastic Output of Image Compression Plan

3.1 Planning for Image Compression

The current implementation of the image compression knowledge in the planner involves selection of the VQ or standard compression algorithms. If the compression technique selected is VQ, knowledge includes codebook selection, vector dimensions and host machine where the compression is executed. In particular, the compression knowledge is incorporated into a general image processing knowledge base for remote sensing data.

Specifically, when the image compression goal is a subgoal of another plan for data archiving, the planner chooses VQ codebooks and vector dimensions based upon user constraints on compression ratios and quality of compressed data. Figure 1 shows an example output from a very simple plan using the operators in the previous section. The interface shows potential resource subscription problems in the bottom two windows, while task intervals for the two steps and the orderings between them are shown in the top window.

We are currently addressing the problem of relaxing user constraints to fit the real-time constraints of ingest. In this case, the planner will continue to relax the compression parameters until both deadlines and resource constraints can be satisfied. To do this, a planning method of interleaving planning and execution will have to be incorporated into the ingest process. For example, progressive VQ requires the application of a particular quality level for the first level of compression to determine the next level's compression ratio. Selection of the codebook at each level must be initiated by the planner as a function of the previous algorithm application.

4 Conclusion

For a first pass, we have shown that Progressive VQ compression can be easily incorporated into the planning process. Because of the time and resource constrained environment of satellite processing, the choice of not only Progressive VQ compression techniques, but also other more traditional approaches, requires the use the coordination between a planner and a scheduler such as PlaSTiC. However, future systems that incorporate an interleaved planning/scheduling approach whereby results are checked during the planning processes are required for the Progressive VQ techniques.

Acknowledgments

The authors would like to thank Mark Boddy (Honeywell TC), Jim White (Honeywell TC), John Beane (Honeywell TC) for their invaluable contributions throughout the project.

References

[1] B.D. Andrews, P.A. Chou, M. Effros, and R.M. Gray. A mean-removed variation of weighted universal vector quantization for image coding. In *Proc. Data Compression Conference, IEEEE Catalog 93TH0536-3*, pages 302 - 309, Snowbird, UT, March 1993.

[2] M. Antonini, M. Barlaud, P Mathieu, and I. Daubechies. Image coding using wavelet transform. *IEEE Trans. on Image Processing*, 1(2):205 - 220, 1992.

[3] C. F. Barnes and E. J. Holder. Classified variable rate residual vector quantization applied to image subband coding. In *Proc. Data Compression (IEEE Catalog Number 93TH0536-3)*, pages 272 - 281, Snowbird, UT, March 1993.

[4] M. Boddy. Temporal reasoning for planning and scheduling. *Sigart Bulletin*, 4(3), July 1993.

[5] M. Boddy, J. White, R. Goldman, and N. Short, Jr. Planning applications in image analysis. In *1994 NASA-GSFC Proceedings of Space Applications of AI*, Greenbelt, MD, May 1994.

[6] D. Civco. Knowledge-based land use and land cover mapping. In *Proceedings ASPRS/ACSM*, volume 3, pages 276 - 91, 1989.

[7] R. F. Cromp, W. J. Campbell, and N. M. Short, Jr. An intelligent information fusion system for handling the archiving and querying of terabyte–sized spatial databases. In *International Space Year Conference on Earth and Space Science Information Systems.* American Institute of Physics, 1992.

[8] B. R. Epstein, R. Hingorani, J. M. Shapiro, and M. Czigler. Multispectral klt-wavelet data compression for landsat thematic mapper images. In *Proc. Data Compression (IEEE Catalog Number 92TH0436-6)*, pages 200 - 208, Snowbird, UT, March 1992.

[9] S. Ghosh, J. Hendler, S. Kambhampati, and B. Kettler. Common lisp implementation of nonlin user manual. Technical report, University of Maryland, Collage Park, MD, February 1991.

[10] R. M. Gray. Vector quantization. In *IEEE ASSP Magazine*, pages 4 - 28, April 1984.

[11] F. Kossentini, W. C. Chung, and M. J. T. Smith. Low bit rate coding of earth science images. In *Proc. Data Compression - IEEE Catalog Number 93TH0536-6*, pages 371 - 380, Snowbird, UT, March 1993.

[12] A. Lansky. A data analysis assistant. In *Proceedings of the 1994 AAAI Spring Symposium on Software Agents*, Stanford, CA, 1994.

[13] M. Manohar and J. C. Tilton. Progressive vector quantization of multispectral image data using a massively parallel simd machine. In *Proc. Data Compression (IEEE Catalog Number 92TH0436-6)*, pages 181 - 190, Snowbird, UT, March 1992.

[14] M. Manohar and J. C. Tilton. Progressive vector quantization on a massively parallel simd machine with application to multispectral image data. submitted to IEEE Transactions on Image Processing, 1994.

[15] T. Markas and J. Reif. Multispectral image compression algorithms. In *Proc. Data Compression - IEEE Catalog Number 93TH0536-6*, pages 391 - 400, Snowbird, UT, March 1993.

[16] D. Mckeown, W. Harvey, Jr., and J. McDermott. Rule-based interpretation of aerial imagery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-7(5):570 - 85, September 1985.

[17] B. Nicolin and R. Gabler. A knowledge-based system for the analysis of aerial images. *IEEE Transactions on Geoscience and Remote Sensing*, GE-25(3), May 1987.

[18] Pennebaker and Mitchell. *JPEG: Still Image Data Compression Standard.* Van Nostrand Reinholt, New York, 1993.

[19] R. F. Rice and J. R. Plaunt. Adaptive variable length coding for efficient compression of spacecraft television data. In *IEEE Trans. on Communication Technology*, pages 889 - 897, 1971. COM-19(I).

[20] R. F. Rice, P. S. Yeh, and W. H. Miller. Algorithms for a very high speed universal noiseless coding module. Publication 91–1, Jet Propulsion Laboratory, Pasadena, California, February 15 1991.

[21] N. Short, Jr. A real-time expert system and neural network for the classification of remotely sensed data. In *Technical Papers, 1991 ACSM-ASPRS Annual Convention*, volume 3. ASCM/ASPRS, 1991.

[22] N. M. Short, Jr. and L. Dickens. Automatic generation of products for terabyte-size geographic information systems using planning and scheduling. *accepted for publication in the International Journal of Geographic Information Systems*, 1994.

[23] J. C. Tilton, M. Manohar, and J. A. Newcomer. Earth science data compression issuess and activities. In *Remote Sensing Reviews*, 1994. 8.

[24] S. Vere. Planning in time: Windows and durations for activities and goals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(3):246 - 267, 1983.

[25] J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Tran. Information Theory*, IT-23:337–343, 1977.