Defence Science Journal, Vol. 64, No. 6, November 2014, pp. 536-541, DOI : 10.14429/dsj.64.8115

Inter-level Spatial Cloud Compression Algorithm

Pranab Kumar Das Gupta[*], Sabyasachi Pattnaik[#], and Manojranjan Nayak[!]

[*]Proof and Experimental Establishment, DRDO, Chandipur-756 025, India
[#]Fakir Mohan University, Balasore -756 019, India
[!]Siksha 'O' Anusandhan University, Bhubaneswar, India
[*]E-mail: pkdasgupta@gmail.com

ABSTRACT

Static images of colour clouds play an important role to predict weather conditions to schedule proof and trial activities, and deploying resources at firing locations and observation posts. In this paper, a new inter-level cloud compression architecture and algorithm has been presented. Distributed architecture suitable for cloud computing has been suggested to implement inter-level compression algorithm (ILCA). Different possibilities between two successive cloud images have been combined to save the images of interest for further processing, ignoring the redundant ones. Vector quantisation technique is briefly discussed to achieve high level of compression. The algorithm presented in this paper can be easily modified to store flying, floating, and moving objects in air, water, and surface, respectively with high level of compression in various defence applications.

Keywords: Cloud compression, edge detection, distributed architecture, vector quantisation, cloud computing

1. INTRODUCTION

Clouds are responsible for Earth's heat balance and hydrological cycle and these happen to be one of the least understood phenomena. These cause large uncertainties in climate models and predictions. Large-scale cloud information is available from earths surface as well as from satellites. Ground-based imaging devices are commonly used to support satellite studies. Recently, increasing development of whole sky imagers enable temporal and spatial high-resolution sky observations[11]. The colour images of clouds are obtained by placing array of cameras at different geographical locations with centralised control and data warehouse for continuous sky observations. Sky imagers are commercially available components and are designed to be location-independent, which can be connected through appropriate communication media with the central server and run under adverse weather conditions. The basic component is a rugged digital camera equipped with an appropriate lens to provide desired field of view. To obtain a high temporal resolution, the cameras are programmed to acquire images at required intervals. It is required to tag the images into 'new', 'shift', 'clear' or 'cloudy' prior to computation and to predict weather conditions. Frequency of image acquisition varies from one image per second to one image per hour (or more) depending on the weather conditions. This phenomenon results in huge collection and creation of image data warehouse. Automation of cloud imaging is difficult to achieve, because of the above said varied change in the weather conditions. If the image-grabbing camera's frequency is set to low, say every 15 seconds, on each day, the number of images will be 5760. Even if the size of image is restricted to 200 x 300 pixels, image size per day will be of the order of 1 GB. For an area corresponding to the radius 1000 km, placement of 100 frame-grabbing cameras is justified. Thus for one day, volume of the data size will be of the order 100 GB! The major issue is to transmit the data to central server and subsequent systematic storage. One solution to reduce this huge volume is to increase the interval between frames from 1 frame per 15 second to, say, 1 frame per hour. However with this strategy, any significant change within an hour on any particular day will be missed out. This will also restrict future studies related to prediction of the weather conditions based on the archived data. Particularly during pre-cyclone, movement of cloud is extremely fast, and to study the profile for future analysis and prediction, every second imaging is essentially required. In such type of condition, storage increases to 1500 GB per day, a huge value!

In this paper, concepts related to edge detection and vector quantisation have been used to conceptualise inter-level compression algorithm (ILCA). ILCA is proposed to reduce the over all data size using successive comparisons of cloud images. In actual cases, colour cloud frames remain unchanged for significantly long durations. Frames, which change over time, must only be stored ignoring the repetitive images. However at times, change of position of cloud is extremely fast. In such type of condition, frame of cloud image must not be missed out and should be grabbed every second from each weather station.

2. LITERATURE SURVEY

A scene is a collection of adjacent shots focusing on the

Received 22 August 2014, online published 13 November 2014

same objects in the same place and at the same time. Cameras with scene detection feature are commercially available. However probably because of the proprietary issue, literature relevant to the same is sparingly available. Review of image and video indexing techniques is reported by Idris & Panchanathan[12]. Majority of the compression-related work associated with moving object is based on video. In the paper published by Andriani & Calvagno[1] redundancy in a colour video sequence for lossless compression is presented. Automatic moving object extraction in video is discussed in the paper published by Haifeng Xu[9]. Significant amount of work related to the vector quantisation and clustering is done by Dubes & Jain[5], Gersho[6], Gray[8], Makhaul[17] and Nasrabadi & Robert[18]. The problem of image data compression is often referred to as low bit-rate coding. The primary design objective is to minimize the average number of bits used to represent a given image sequence in digital form. In the paper published by Patnaik[15], LBG Algorithm has been improved to deal with colour cloud image[15]. Selection of Training Set and Initial Codebook is an important feature to obtain the final Codebook in less number of iterations with minimal mean square error, and the same is achieved by elimination of redundant information that the human visual system cannot perceive. In the said paper, this feature is exploited to form Training Set and Initial Codebook using histogram to achieve intra-level compression of the order of 40 with low mean square error and visual distortion[19]. In the papers by Chen[2]and Inmon[13] data mining concepts from database perspective is dealt in detail. Compression, clustering and pattern discovery in very high dimensional discrete attribute data sets is discussed by Koyuturk[14]. Data warehouse and data mining related concepts are available in the classical book by Han & Kamber[10]. Distributed architecture along with appropriate use of client/server and three-tier is illustrated by Das Gupta[3]. Dot Net Architecture and ASP. Net to develop the logic of ILCA is available in the book published by Das Gupta[4]. Detailed treatment related to digital image processing and data compression is available in Gonzalez & Woods[7] and Sayood[20], respectively. Current trends related to cloud computing are discussed in the paper written by Schneiderman[21]. Content-based image retrieval is discussed in the paper by Hanmandlu[22].

3. ILCA IMAGE ACQUISITION ARCHITECTURE

Three-tier architecture relevant to ILCA is suggested in Fig. 1. A typical ILCA scenario from networking perspective is shown in Table 1.

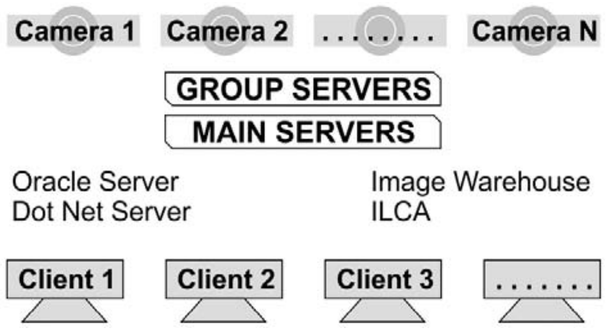

Figure 1. ILCA architecture.

In Fig. 1, cameras at the top are equipped with storage and transmission device. In the centre, Group-Level Servers (for logic) and Main-Level Servers (for backend) are shown. It is worthwhile to mention that cluster of servers at both the levels are required to process the acquired images concurrently. Hence, blade servers with storage area network (SAN) may be used for the purpose. The overall processing needs to handle simultaneous images gathered from multiple cameras to select the images of interest using ILCA. Group servers associated with the cameras are responsible for processing at inter level compression of images. Group-Level and Main-Level servers are connected in mesh type of network architecture and the same is shown in Fig. 2.

Oracle Database Server is used in the data-tier (or backend). In the logic tier (or middle tier) Dot Net server is configured to cater to the computing need of ILCA. Oracle Client is installed in the middle tier so that the software can connect to Oracle Database Server. In this setup, one server each is used in data tier and logic tier. In the presentation layer, one or all the clients can access the selected images using the IP address 192.168.128.200 of the logic tier server. Logic tier on the other hand, communicates with the database server using the IP address 192.168.128.100. The architecture suggested above can be easily used to setup a private cloud.

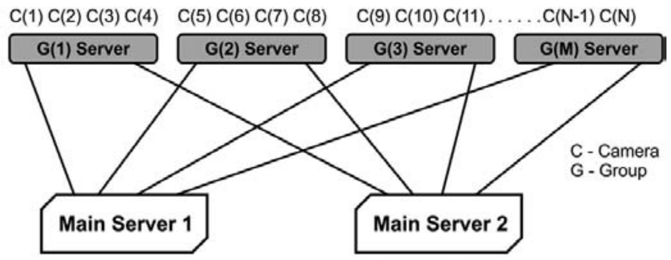

Figure 2. ILCA group and main servers connectivity.

Table 1. ILCA network scenario

Features	Data tier (Backend)	Logic tier (Middle tier)	Presentation tier (Frontend)
Operating system	Windows 2008 Server	Windows 2008 Server	Windows XP Prof
Network protocol	TCP/IP	TCP/IP	TCP/IP
IP address	192.168.128.100	192.168.128.200	192.168.128.1 to 50
Machine name	BackEnd	MiddleTier	FrontEnd1 to FrontEnd50
Software	Oracle database	Dot Net Oracle Client ILCA	Internet Explorer (IE)

In Fig. 2, it is assumed that cameras C(1), C(2), … , C(N) are deployed in M geographical locations. G(1), G(2), … , G(M) Servers are equipped with Oracle database server and Dot Net technology and are responsible for selection of useful images using ILCA. Main Servers1 and 2 are basically cluster of servers responsible for storage of selected images (through ILCA). Finally in the front-end, clients can access the images of interest selected through ILCA for further processing. The architecture suggested can also be realised through private clouds with virtualisation. This will ensure handling of security issues prevalent in public cloud.

4. INTER-LEVEL COMPRESSION ALGORITHM

ILCA assumes that a camera is working in the respective location and capturing images at the desired intervals. Previt, Sobel, and Canny edge detection algorithm was found suitable for ILCA. Sobel and Previt algorithms were found computationally least expensive but highly sensitive to noisy image. Canny algorithm on the other hand, is computationally expensive but most suitable to deal with noise. Mask used by all the above-mentioned algorithms is 3x3. Appropriate algorithm may be chosen as per the input image. A good deal of comparison on the edge detection algorithms is available in the paper published by Maini & Aggarwal[16]. ILCA works on two successive cloud images to decide whether to store or ignore the image. Variables associated with ILCA are as under.

I	Image	E	Edge
T	Tolerance Limit	H	Histogram Mean
c	Counter (= 1)	pp	Pixel Position
pc	Pixel Count	pv	Pixel Value
cs	Clear Sky Pixel		
T_{pc}	Tolerance corresponding to pixel count		
T_{pp}	Tolerance corresponding to pixel position		
T_{cs}	Tolerance corresponding to clear sky pixel		
T_{pv}	Tolerance corresponding to pixel value		
Tags	New, shift, clear, ucloudy		

There are the following four possibilities between two successive images in two frames.

- No change in the position of both the images of the cloud.
- Change in the position of both the images of the cloud.
- Different images of cloud in both the frames.
- Uniformly cloudy or clear image of cloud in the frame.

Each of the above-mentioned possibilities are elaborated in succeeding sections.

4.1 ILCA Possibility 1

No change in the position of both the images of the cloud

This possibility is prevalent during summer and winter conditions, which result in high level of compression.

Detect and store the edge for the first image (i.e., I_c). Find the number of pixels associated with the edge. Store I_c with tag 'New'.

[Acquire I_c]
[Detect $E(I_c)$]
[Save $I_{c(new)}$]

For the second image (i.e.,I_{c+1}), navigate the edge detection mask (3x3 matrix) using the stored edge path corresponding to I_c. This will save lot of computation time. Edge coordinates and number of pixels will be the same as there is no change in position of both the images of the cloud. Ignore I_{c+1} and increment the Image Counter. T_{pp} and T_{pc} are required to allow tolerance.

[Acquire I_{c+1}]
[Detect $E(I_{c+1})$]
[Ignore I_{c+1}]
[c++]

4.2 ILCA Possibility 2

Change in the position of both the images of the cloud

This possibility exists throughout the year. Shift of cloud in the same frame contributes to significant number of frames. Hence, in such type of possibility low compression is achieved. To achieve good amount of compression, vector quantisation is suggested. Instead of transmitting the full image, index along with the edge coordinates and rotation may be transmitted. This will lead to significant increase in overall compression.

Detect and store the edge for the first image (i.e., I_c). Find the number of pixels associated with the edge. Store I_c with tag 'New'.

[Acquire I_c & I_{c+1}]
[Detect $E(I_c)$ & $E(I_{c+1})$]
[Save$I_{c(new)}$]

For the second image (i.e., I_{c+1}), navigate the mask (3x3 matrix) using the stored edge path corresponding to I_c. As both the images are not in the same position, edge in both the images will be in different positions. However, the number of pixels associated with the edge in both the images will be the same. This indicates that there is a shift of cloud in I_{c+1}. Store I_{c+1} with tag 'Shift' and increment the Image Counter. In this case, T_{pp} and T_{pc} are essentially required to allow tolerance.

[Acquire I_c & I_{c+1}]
[Detect $E(I_c)$ & $E(I_{c+1})$]
[Save$I_{c+1(shift)}$]
[c++]

4.3 ILCA Possibility 3

Different images of cloud in both the frames

This possibility is prevalent during rainy season. In such type of possibility, high speed computing and fast communication is desired so that useful images are not missed due to data crunching and/or congestion in network. In such type of possibility, very low compression is achieved.

Detect and store the edge for the first image (i.e., I_c). Find the number of pixels associated with the edge. Store I_c with tag 'New'.

[Acquire I_c]
[Detect $E(I_c)$]
[Save $I_{c(new)}$]

For the second image (i.e., I_{c+1}), navigate the mask (3x3 matrix). As both the images are different, edge in both the images will also be different. Number of pixels associated with both the edges will also be different. This ensures that I_{c+1} is a new image. Store I_{c+1} with tag 'New' and increment the Image

Counter. In this case also, T_{pp} and T_{pc} are essentially required to allow tolerance relevant to pixel position and pixel count respectively.

[Acquire I_{c+1}]
[Detect $E(I_{c+1})$]
[Save $I_{c+1}(new)$]
[c++]

4.4 ILCA Possibility 4

No image of cloud in both the frames

This possibility is prevalent during summer and winter condition, which results into high level of compression.

Apply Edge Detection Algorithm on the first image (i.e., I_c). As the sky is either clear or cloudy, edge will not be formed.

[Acquire I_c]
[Detect $E(I_c)$]

Confirm the same by generating a histogram corresponding to the pixels. Number of elements will be 1 to 3, each with large frequency. This will confirm that either the sky is clear or uniformly cloudy. Store I_c with tag 'Clear' or 'uCloudy'. Incorporate T_{pv} and T_{cs} to allow tolerance.

[If $H(I_c) = H_{cs}$+(-) T_{cs} then Save $I_{c(clear)}$ else Save $I_{c(ucloudy)}$]

As the sky is clear (or uniformly cloudy), edge will not be generated. Confirm the same through histogram. Ignore I_{c+1} and increment the Image Counter.

[Ignore I_{c+1}]
[c++]

4.5 ILC Algorithm

In realistic scenario, image in second frame is not known. To handle the same, ILCA combined all the four possibilities to store frame of interest, filtering out the redundant ones. ILCA is shown as follows.

```
Begin
    $c$=1
    Loop
        Acquire $I_c$&$I_{c+1}$
        If $E(I_c)$ detected then
            Find $E(I_{c+1})$
            If $E(I_c)_{pc} = E(I_{c+1})_{pc}$ +(-) $T_{pc}$ then
                If $E(I_c)_{pp} = E(I_{c+1})_{pp}$ +(-) $T_{pp}$ then
                    Save $I_{c(new)}$
                    Ignore $I_{c+1}$
                Else
                    Save $I_{c(new)}$
                    Save $I_{c+1(shift)}$
                End if
            Else
                Save $I_{c(new)}$
                Save $I_{c+1(new)}$
            End if
        Else
            If $H(I_c) = H_{cs}$+(-) $T_{cs}$ then
                Save $I_{c(clear)}$
            Else
                Save $I_{c(cloudy)}$
            End if
            If $H(I_c) = H(I_{c+1})$+(-) $T_{pv}$ then
                Ignore $I_{c+1}$
            Else
                Save $I_{c+1(new)}$
            End if
        End if
        If no image then
            Exit
        Else
            $c$++
        End if
    End Loop
End
```

5. ILCA TEST RESULT

ILC Architecture as well as ILC algorithm have already been discussed in detail in the previous sections. In this section, a typical scenario of selection and rejection of cloud images is demonstrated. In this, out of 200 images, only 8 images (3A to 3H) are found useful, resulting into a compression of 25. Table 2 consolidates the images with relevant tags. It may be noted that compression of the order 1000-2000 is possible during summer and winter. On the contrary, compression may be even zero during volatile cyclonic weather. ILCA can handle both the extreme conditions efficiently. Fig 3.

ILCA will result in saving useful images as shown in Table 2.

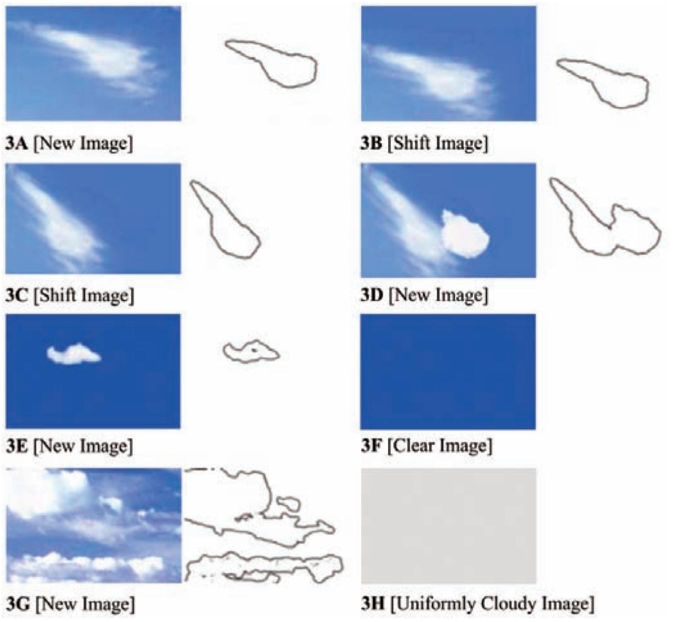

Figure 3. ILCA test result.

6. CONCLUSION

In this paper a new algorithm, viz., ILCA along with its architecture is presented. In case of actual implementation, sequencing along with time stamping and creation of image warehouse are important issues and these need to be addressed. This enables unique identification of each

Table 2. ILCA image data

Image Counter	Image	Tag
1	3A	New
2	3B	Shift
3	3C	Shift
50 images similar to 3C ignored		
4	3D	New
70 images similar to 3D ignored		
5	3E	New
30 images similar to 3E ignored		
6	3F	Clear
50 images of clear sky ignored		
7	3G	New
8	3H	uCloudy

image for further processing. Concurrency control also plays an important role, especially during monsoon, when large number of simultaneous images are transmitted from different weather stations to the group server and the main server. The paper is multi-field in nature and all the relevant topics have been briefly covered with relevant references.

7. ACKNOWLEDGEMENT

Authors are thankful to Shri R Appavuraj, Director, Proof & Experimental Establishment, Defence R&D Organisation for his encouragement and kind support.

REFERENCES

1. Andriani, S. & Calvagno, G. Lossless compression of colour sequence using optimal linear prediction theory. *IEEE Trans. Image Process.*, 2008, **17**(11), 2102-2111. doi: 10.1109/TIP.2008.2003391
2. Chen, M.S.; Han, J. & Yu, P.S. Data Mining: An overview from database perspective. *IEEE Trans. Knowl. Data Eng.*, 1996, **8**(6), 866-883. doi: 10.1109/69.553155
3. Das Gupta P.K. Database management system, Oracle SQL and PL/SQL. Second Edition, PHI, 2013.
4. Das Gupta P.K. Developing web applications using ASP. Net & Oracle. Ed 2, PHI, 2013.
5. Dubes, R. & Jain, A.K. Clustering techniques: The user's dilemma. *Pattern Recognition*, 1976, **8**(4), 247–260. doi: 10.1016/0031-3203(76)90045-5
6. Gersho, A. Asymptotically optimal block quantization. *IEEE Trans. Info. Theory*, 1979, **25**(4), 373-380. doi: 10.1109/TIT.1979.1056067
7. Gonzalez, R.C. &Woods R.C. Digital image processing using Matlab. McGraw Hill, 2010.
8. Gray, R.M. Vector quantization. *IEEE ASSP Magazine*, 1984, **1**(2), 4 - 29. doi: 10.1109/MASSP.1984.1162229
9. Haifeng, Xu; Younis, A.A. & Kabuka, M.R. Automatic moving object extraction for content based applications. *IEEE Trans. Circuits Syst. Video Technol.*, 2004, 14(6), 796-812.
10. Han, J. & Kamber, M. Data mining: Concepts and techniques. Morgan Kaufmann Publishers, 2002.
11. Heinle, A.; Macke, A. & Srivastav A. Automatic cloud classification of whole sky images. *Atmos. Meas. Tech.*, 2010, **3**(3), 557-567. doi: 10.5194/amt-3-557-2010
12. Idris, F. & Panchanathan, S. A review of image and video indexing techniques. *J. Visual Commun. Image Representation*, 1997, **8**(2), 146-166. doi: 10.1006/jvci.1997.0355
13. Inmon, W.H. The data warehouse and data mining. *Commun. ACM*, 1996, **39**(11), 49–50. doi: 10.1145/240455.240470
14. Koyuturk, M.; Grama, A. & Ramakrishnan, N. Compression, clustering, and pattern discovery in very high-dimensional discrete-attribute data sets. *IEEE Trans. Knowl. Data Eng.*, 2005, **17**(4), 447- 461. doi: 10.1109/TKDE.2005.55
15. Linde, Y.; Buzo, A. & Gray R.M. An algorithm for vector quantizer design. *IEEE Trans. Commun.*, 1980, **28**(1), 84 - 95. doi: 10.1109/TCOM.1980.1094577
16. Maini, Raman & Aggarwal, Himanshu. Study and comparison of various image edge detection techniques. *Int. J. Image Process.*, 2009, **3**(1), 1-11.
17. Makhaul, J.; Roucos, S. & Gish H. Vector quantization in speech coding. *In* Proceedings of IEEE, 1985, **73**(11), 1551 - 1588. doi: 10.1109/PROC.1985.13340
18. Nasrabadi, N.M. & King, R. A. Image coding using vector quantization: A review. *IEEE Trans. Commun.*, 1988, **36**(8), 957 - 971. doi: 10.1109/26.3776
19. Patnaik, S.; Das Gupta, P.K. & Nayak, M. Mining images using clustering and data compressing techniques. *Int. J. Info. Commun. Technol.*, 2008, **1**(2), 131–147. doi: 10.1504/IJICT.2008.019098
20. Khalid, Sayood. Introduction to data compression. Elsevier India, 2010
21. Schneiderman, R. For cloud computing, the sky is the limit. *IEEE Signal Process. Magazine*, 2011, **28**(1), 15-144. doi: 10.1109/MSP.2010.938751
22. Hanmandlu, Madasu & Das, Anirban. Content-based image retrieval by information theoretic measure. *Def. Sci. J.*, 2012, **61**(5), 415-430.

CONTRIBUTORS

Dr Pranab Kumar Das Gupta, PhD, is a Scientist 'F' and Additional Director in PXE, Chandipur. His expertise in development, execution and implementation of projects has immensely helped his organization. He has published more than 20 papers in National and International Journals and Seminars. He has also authored four books.

Dr Sabyasachi Pattnaik (MTech and PhD in Computer Science) is Professor in the Department of Information and Communication Technology at Fakir Mohan University, Balasore, Odisha. His fields of interest are soft computing, bioinformatics and data mining. He has published 65 research articles in national and international journals, and in proceedings/conferences publications. He has also authored one book.

Dr Manojranjan Nayak, (MTech, PhD and DSc in Computer Science) is President and Founder of the Siksha 'O' Anusandhan University, Bhubaneswar, Odisha. Professor Nayak has been an academician par excellence and is well known for his work in the field of soft computing. He has published 70 research articles in national and international journals, and in proceedings/conferences publications.