SPARSE DISTRIBUTED MEMORY PROGRAMMING IN DIGITAL CAMERA USING DIGITAL IMAGE PROCESSING

A.SENTHILRAJAN

Abstract: Digital camera controlled program is a broad area of research. Vision based approaches deserves special attention, inexpensive and powerful. However, problems such as illumination changes and noise are difficult to overcome. A kind of associative memory with high dimensional binary vectors is used, which also exhibits behaviors in many aspects similar to those of the human brain. This paper analyses the impact of using different image processing techniques on the images using histogram equalization and smoothing using Gaussian filter. The result shows that equalization and smoothing have a positive effect on the performance of the system.

Keywords: Image processing, Sparse Distributed Memory, Smoothing, Gaussian filter

Introduction: Many different approaches have been tried to localize and robust way and some of those approaches can only used in structured environments. Since they are based on the recognition of artificial landmarks, beacons or a similar strategy that improve the accuracy of the system but requires special conditions. More generic strategies that work in unstructured environments include mapping and localization using digital cameras.

Vision based approaches are biologically inspired, since human use mostly vision for localization [2]. The sensors required are inexpensive, but the processing power needed is huge. Every single image is usually described by tens, hundreds or thousands of images. The use of cognitive information, in which the raw images are replaced by formal descriptions of the contents of the images, may solve part of the problem, but that is still an open area of research. The images alone are a means for instantaneous localization.

The SDM is a kind of associative memory based on the properties of high-dimensional Boolean spaces and thus suitable to work with large binary vectors such as raw images [4]. The present paper describes the experiments using different image processing techniques, in order to make the system more robust to illumination changes and image noise.

Section 2 explain view sequence, section 3 briefly describes the SDM. In section 4 experimental platform is presented. Section 5 describes the image processing techniques used. Section 6 discusses the results obtained and section 7 draws conclusions and future work.

View Sequences: This approach is based on visual memories stored into a SDM. It stores a sequence of views for each image captured. Images that are very similar to previously stored images are discarded because they would be with high probability by not adding any relevant information to the known facts. Localization is estimated based on the similarity of two views one stored during the supervised learning stage and another grabbed in real-time.

Sparse Distributed Memory: The Sparse Distributed Memory is an associative memory model proposed by Kanerva in the 1980s. It is suitable to work with high-dimensional binary vectors. Kanerva shows that the SDM exhibits the properties of large Boolean spaces, to a great extent. Which are similar to that of the human cerebellum? The SDM naturally implements behaviours such as tolerance to noise, operation with incomplete data, parallel processing and knowing that one knows. In the proposed approach, an image is regarded as a high-dimensional vector, and the SDM is regarded simultaneously as a sophisticated storage and retrieval mechanism and a pattern recognition tool. For space constraints the original SDM model is not described in the present paper. In the present work, a model that we call "auto-associative arithmetic" is used, as exemplified in figure1.The main modules of the SDM are an array of addresses and an array of data vectors. It is even acceptable for auto associative version, in which the same array is used simultaneously as addresses and data, as long as datum ς is only stored at location ς .The auto-associative memory needs about one half of the storage space.

Every input address will activate the memory addresses that are within a predefined activation radius. Different methods can be used for computing the distance–the present implementation uses the sum of the absolute differences.

Reading from the memory is done by averaging the integer values column wise. Learning is achieved by updating each byte value using the equation:

$$h_t^k = h_{t-1}^k + \alpha.(x^k - h_{t-1}^k), \qquad \alpha \ni \square \wedge 0 \le \alpha \le 1 \qquad (1)$$

In the equation, h_t^k is the k^{th} number of the memory location, at time t, x^k is the corresponding number in the input vector x and α is the learning rate. In the present implementation α was set to 1, enforcing one learning.

The memory locations are managed using the Randomized Reallocation (RR) algorithm. Using the RR, the system starts with an empty memory and allocates one locations when there is a new datum which cannot be stored into enough existing locations. The new locations are placed randomly by the neighborhood of the new address.

Experiment Platform: The software architecture contains three basic modules a) the SDM, where the information is stored, b) the Focus and c) an interface and image processing. The acquired image is processed using the IDL program. IDL (the Interactive Data Language) is a complete computing environment for

the interactive analysis and visualization of data. IDL integrates a powerful array-oriented language with numerous mathematical analysis and graphical display techniques. Programming in IDL is a time-saving alternative to programming in FORTRAN or C. Using IDL, tasks which require days or weeks of programming with traditional languages can be accomplished in hours. The data can explore interactively using IDL commands and then create complete applications by writing IDL programs. Many numerical and statistical analysis routines, including Numerical Recipes routines are provided for analysis and simulation of data. Compilation and execution of IDL commands provide instant feedback and hands-on interaction. Operators and functions work on entire arrays (without using loops), simplifying interactive analysis and reducing programming time. IDL's flexible input/output facilities allow us to read any type of custom data format.

Adaptive Histogram equalization: HISTOGRAM can optionally return an array containing a list of original array subscripts that contributed to each histogram bin. This list, commonly called the reverse (or backwards) index list, efficiently determines which array elements are accumulated in a set of histogram bins. A typical application of the reverse index list is a reverse histogram or scatter plot interrogation—a histogram bin or 2D scatter plot location is marked with the cursor and the original data items within that bin are highlighted.

Adaptive histogram equalization involves applying equalization based on the local region surrounding each pixel. Each pixel is mapped to an intensity proportional to its rank within the neighborhood. This type of equalization also tends to reduce the disparity between peaks and valleys within the image's histogram.

```
PRO Adaptive Equalizing
Import the image from the file
file = FILEPATH('image.png', $   SUBDIRECTORY = ['examples', 'data'])
image = READ_PNG(file, red, green, blue)
ImageSize = SIZE (image, /DIMENSION)
Initialize the display.
DEVICE, DECOMPOSED = 0
TVLCT, red, green, blue
Create a window and display the original image.
WINDOW, 0, XSIZE = Image Size[0], YSIZE = Image Size[1], $
TITLE = 'Original Image'
TV, image
Create another window and display the histogram of the
The original image.
WINDOW, 1, TITLE = 'Histogram of Image'
```

```
PLOT, HISTOGRAM (image), /XSTYLE, /YSTYLE, $
TITLE = ' Image Histogram', $
XTITLE = 'Intensity Value', $
YTITLE = 'Number of Pixels of That Value'
Histogram-equalize the image.
Equalized Image = ADAPT_HIST_EQUAL(image)
Create another window and display the equalized image.
WINDOW, 2, XSIZE = Image Size[0], YSIZE = Image Size[1], $
TITLE = 'Adaptive Equalized Image'
TV, equalized Image
Create another window and display the histogram of the
equalized image.
WINDOW, 3, TITLE = 'Histogram of Adaptive Equalized Image'
PLOT, HISTOGRAM(equalized Image), /XSTYLE, /YSTYLE, $
TITLE = 'Adaptive Equalized Image Histogram', $
XTITLE = 'Intensity Value', $
YTITLE = 'Number of Pixels of That Value'
END
```

Figure 1: Original

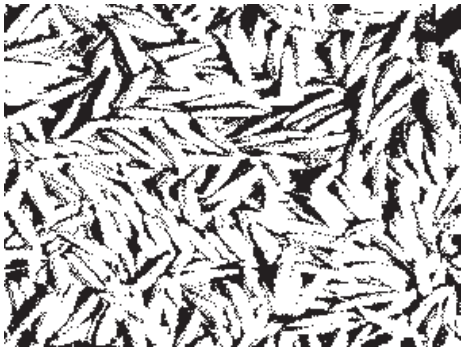

Figure 2: Normalised

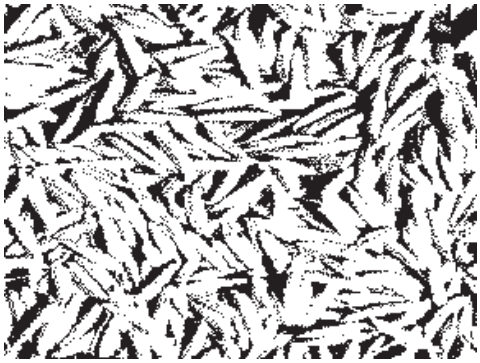

Figure 3 : Smoothed with Equalised

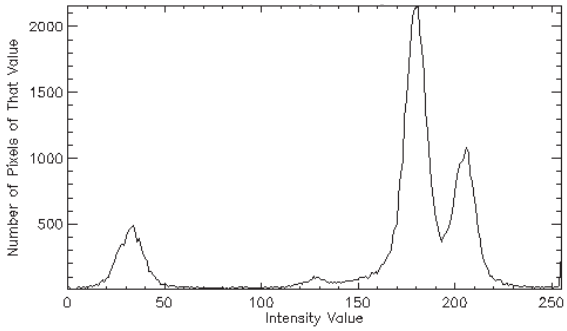

Figure 4: Original image histogram

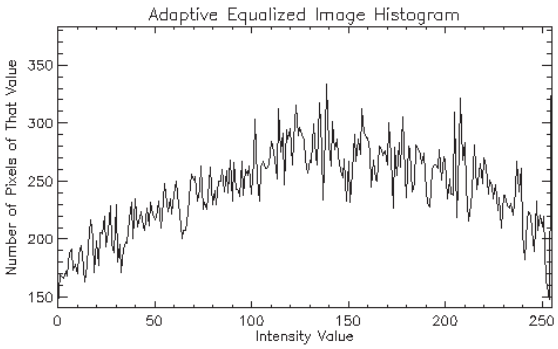

Figure 5: Smoothed equalised histogram

Smoothing: Gaussian filters are often used in image processing as a way to reduce noise levels. The final image is blurred, and the contours are smoother than they were in the original image. The process consists of running through the image a filter in which each pixel is replaced by a weighted average of the neighboring pixels, where the nearest neighbors contribute more than farthest neighbor, according to the Gaussian function. For an image, a two dimensions Gaussian must be applied, as shown in Equation below.

$$G(x,y) = \frac{1}{2\pi\alpha^2} e - \frac{x^2 + y^2}{2\sigma^2}$$

In the formula, σ is the standard deviation of the Gaussian distribution. A large σ will give more importance to farther pixels, while a small σ gives more importance to the nearest pixels. In practice, pixels outside the radius 3σ are usually ignored, for their contribution to the final value is negligible.

In the present work, the Gaussian filter was applied using IDL. The standard deviation was computed automatically, but only the contributions of the immediate neighbors were used.

Figure 7 show the original image, an equalized image and a smoothed image. Obviously, the same image can be subject to different processing methods. For example, it makes sense to smooth an image and then equalize. It makes no sense to equalize and contrast stretch, since both methods intend to achieve the same goal and contrast stretch will have no effect on an equalized image.

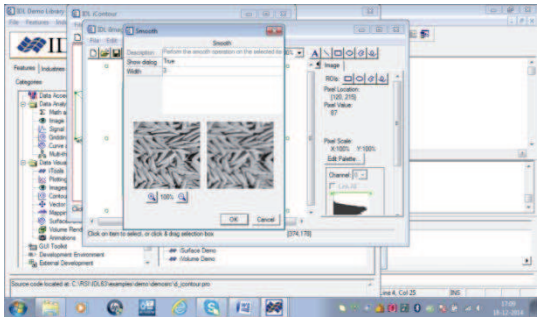

Figure 6: output generated by IDL window

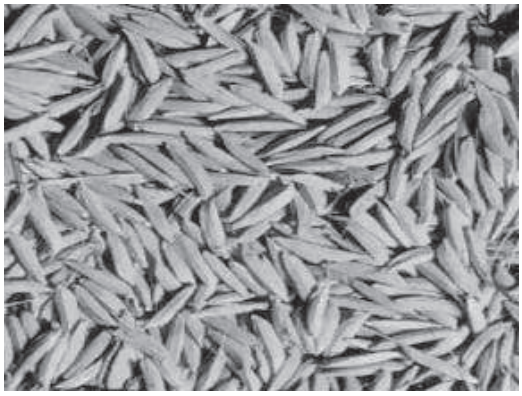

Figure 7: Smoothed image

iTools Statistics
Original Image
Image Planes
Dimension: [288, 216] (62208 elements)
Mean: 147.689
Total: 9.18746e+006
Minimum: 10
at: [277, 203]
Maximum: 238
at: [263, 5]
Variance: 2399.04
Standard Deviation: 73.9484
Absolute Deviation: 64
Skewness: 0
Kurtosis-3: -1.20472
Result:
Smoothed Image
Image Planes

Dimension: [288, 216] (62208 elements)
Mean: 147.739
Total: 9.19057e+006
Minimum: 12 at: [171, 150]
Maximum: 235 at: [263, 5]
Variance: 2360.02
Standard Deviation: 48.5801
Absolute Deviation: 40.5493
Skewness: -0.769195
Kurtosis-3: -0.480333

Discussion: As the statistics result shown, the original image standard deviation has nearly 73%, whereas the smoothed image after applying the image processing techniques the standard deviation is reduced to 48% which shows that the better result and reduced the noise from the image. Equalization has a very large impact on the average distances. That happens because the pixel values are stretched over all the domains for histogram equalization. Smoothing the images using the Gaussian filter greatly reduces the average distances. That is the pixel values are approximated to the values of the neighboring pixels. The performance of the system is excellent with smoothing.

Conclusion: Sparse Distributed Memory programming in Digital camera using Digital image processing is a long sought goal. However, there are many problems to be solved, such as the presence of noise in the images and illumination changes. The approach followed in the present work relies on memories stored into SDM. The performance of the system is improved by processing the images using a Gaussian filter and histogram equalization. Contrast normalization usually produces bad results. Further research will be done in order to identify relevant features in the images. The vectors stored into the SDM will contain features information, leading the way to work at a cognitive level. That should improve the performance of the system and possibly decrease memory space and processing time.

References:

1. Pentti Kanerva, Sparse Distributed Memory, MIT Press Cambridge,1988
2. Mateus Mendes, Manuel M.Crisostomo, and A.Paulo Coibra. Robot navigation using a sparse distributed memory. In Pro.of IEEE Int. Conf. on Robotics and Automation, Pasadena, California, USA. May 2008.
3. Mateus Mendes, Manuel M.Crisostomo, and A.Paulo Coimbra. Assessing a sparse distributed memory using different encoding methods. In Proc. Of World congress on Engineering, London, UK, July 2009.
4. *Raghvendra Upadhyay, Goutham Dutta, Mohit Kumar Singh, Dalip Bera, Neelesh Srivastava,* A Numerical Solution of the Point Kinetics Equations and Its

Validation; Mathematical Sciences international Research Journal ISSN 2278 – 8697 Vol 3 Spl Issue (2014), Pg 993-1005

5. Bohdana Ratitch and Donia Precup. Sparse distributed memories for on-line value-based reinforcement learning. In ECML, 2004
6. Dr.E.Ramaraj, A.Senthilrajan, "Parallel Sorting Algorithm", 12th International workshop on Future Trends of Distributed Computing Systems, Kunming, China.21-23, October 2008. Paper presented and published.
7. Dr.E.Ramaraj, A.Senthilrajan, "Multi Core processor to support network parallel Image processing applications", International conference on signal processing system 15-17 May,2009, Singapore, Organized by International association of computer science and IT. IEEE computer society, ISBN: 978-0-7695-3654-5. Paper presented and published.
8. Dr.E.Ramaraj, A.Senthilrajan, International multi conference of engineers and computer scientists, 17-19, March,2010, "High Density Impulse Noise removal using Median Filter", Hongkong, International Association of Engineers. ISBN: 978-988-18210-4-1, ISSN: 2078-0958. Paper presented and published.
9. S. Sudha,R. Alphonse Santhanam, Double Domination on Generalized Petersen Graphs; Mathematical Sciences International Research Journal ISSN 2278 – 8697 Vol 3 Issue 1 (2014), Pg 8-10
10. Dr.E.Ramaraj, A.Senthilrajan, "Median Filter Using Multiprocessing in Agriculture", in 10th International Conference on Signal Processing (ICSP'10), Oct.24-28, 2010, Beijing, China, Organized by IEEE Signal Processing Society Paper was presented and published.
11. Dr.A.Senthilrajan,"Paddy Grade and Dirt classification using Image Processing Techniques", in C2SPCA2013Oct 10,11,2013, IEEE conference in Bangalore, India. Paper was presented and published.
12. Dr.E.Ramaraj, A.Senthilrajan, "Median Filter in Agriculture", in World Congress on Engineering and Computer Science 2010 San Francisco, USA, 20-22 October, 2010 Organized by the International Association of Engineers (IAENG). ISBN: 978-988-17012-0-6. Published.
13. Kankeyanathan Kannan, Discrete Group With the Twisted Rapid Decay; Mathematical Sciences international Research Journal ISSN 2278 – 8697 Vol 3 Issue 2 (2014), Pg 516-518
14. Dr.E.Ramaraj,A.Senthilrajan,"Filteringtechniques",inNationallevelconference-2006atJyotinivascollege,Koramangala,Bangalore.Paperwaspresented.

A.SenthilRajan, Computer Centre, Alagappa University,Karaikudi,Indiaagni_senthil@yahoo.com