Feature Extraction in Video sequence using Self-Organizing Map

Deepak D. Shudhalwar,
Assistant Professor
Department of Engineering & Technology, PSSCIVE,
Shyamala Hills, Bhopal (MP), India
Email-id: dipakds@yahoo.com

Abstract-**In this paper, we worked for compressing the video sequence using Self-organizing map (SOM) for compressed video sequence. Self-organizing map is an efficient method for reducing the size of video frames in the neural network field. The SOM algorithm is based on unsupervised, competitive learning. It extracts the useful features from video frames by feature extraction method. In our approach, the video sequence is first divided into video frames. The frames are processed by passing to the input of SOM that outputs the informative features by removing the redundant information contained in video frames. The codewords are generated using SOM that can be used for reconstruction of the encoded video frames by using the same codeword at the receiver side.**

I. Introduction

The need for effective video compression is evident in almost all applications where storage and transmission of video frames are involved. Video compression is mainly focused on reducing the number of bits needed to represent video frames. There are two basic types of compression: lossless compression and lossy compression. In lossless compression schemes, there is perfect data matching of the reconstructed video with the original video and the reconstructed video frames is identical to the original video frames. Lossy schemes are capable of achieving much higher compression. Compression ratio is simply the size of original video frame divide by the size of the compressed one. The aim of video compression is to develop a scheme to encode the original video image I into the fewest number of bits such that the video image I' reconstructed from this reduced representation through the decoding process is as similar to the original video image as possible [1, 2]. There are many applications in which this technique is useful as for videoconferencing, e-learning, medical imaging, digital libraries, and many other areas [2, 3].

Self-organizing maps (SOMs) are a data visualization technique invented by Professor TeuvoKohonen which reduces the dimensions of data through the use of self-organizing neural networks. The unsupervised learning feature of SOM is used for reducing dimensions by producing a map of usually one or two dimensions which plot the similarities of the data by grouping similar data items together. Kohonen's SOM provides a topology preserving mapping that is a mapping from high dimensional space onto a plane [4, 5, 6].

The Java Media Framework (JMF) is a versatile package used for handling the audio and video files in Java during the video transmission. Java Media Framework API is important for building multimedia application which can capture, play, stream and transcode multiple media formats that allows the development of cross-platform multimedia applications. The Real-time Transport Protocol (RTP) provides end-to-end delivery services for data with real-time characteristics, such as interactive audio and video or simulation data, over multicast or unicast network services. The RTP defines a standardized packet format for delivering audio and video over the Internet and can carry any data with real-time characteristics [15, 16].

II. Proposed Work

In this paper we worked over the video sequence, video is compressed using Self-Organizing Map (SOM) that outputs the compressed feature vector which is further transmitted over the network. The similar feature vectors at the output of SOM neural network are grouped into custers. In our approach, the video sequence is divided into video frames depending on user defined frame rate of video sequence. The video frames obtained are stored in memory after preprocessing the video. The RGB video frames are converted into gray scale video frames and those frames are then applied to the input of the self-organizing map (SOM). The SOM is an efficient technique for feature extraction from the video frames which outputs the informative features of the video frames. The size of the feature vector or the SOM grid size depends on the number of neurons defined by the user during the initialization of SOM. The SOM is used for dimensionality reduction, thus the dimension of video frames are reduced by exracting the useful features.

The codebook is generated using the feature vectors coressponding to each video frame
This paper is organized as follows. The next section presents the simulation design and implementation details of the problem. SOM algorithm is introduced in Section 3.2.1, Clustering is discussed in Section 3.3 and Multimedia Synchronization is discussed in Section 3.4. Section 3.5 describes the Java Media Framework (JMF) and Real-time Transport Protocol (RTP). Section 4 describes the entire processing steps of this research. The experimental results and discussions are presented in Section 5. Section 6 concludes this study and gives some directions for future research.

III. Simulation Design and Implementation Details

This section describes the experiments of extracting video frames and then used them as an input to the self-organizing map (SOM). The SOM outputs the feature vector corresponding to each video frame in the video sequence. The outputs of the SOM are compared and common feature vectors are found which are then grouped to form clusters.

A. Set of video frames used as an input to the SOM

In our approach, the video frames extracted from 1 second video used for the simulations are shown in Figure 1. Each video frame consists of 120x160 pixel matrix and we have considered frame rate as 15 frames per second that is user defined. The video frames are converted from RGB format to gray scale format. The gray scale video frames are used for further processing.

The set of video frames x_1, x_2, x_3……….x_{15} are preprocessed and stored in memory in gray scale format. The video frames data are loaded in memory and then applied to the input of Self-organizing map. A brief discussion of self-organizing map is given in the next section.

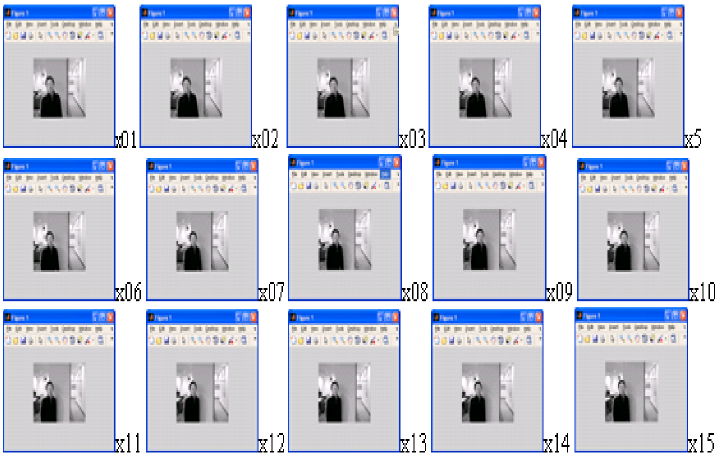


Figure 1: Set of video frames

The video frames are converted from RGB format to gray scale format. The gray scale video frames are used for further processing. The set of video frames x1, x2, x3……….x15 are preprocessed and stored in memory in gray scale format. The video frames data are loaded in memory and then applied to the input of Self-organizing map. A brief discussion of self-organizing map and Hopfield neural network is given in the next section.

B. Mathematical Modeling

Initially the set of video frames x1, x2 …….x15 are preprocessed using following function. Each video frame is R X S matrix that is passed through mplay() function.

F [mplay $(x_i)_{RXS}$] = $[f_i]_{R\ X\ S}$
for all i where i= 1to 15. {RGB video frames}

F [$(f_i)_{RXS}$] = G $[(f_i)_{RXS}]$
for all i where i= 1 to 15. {Gray scale frames}

These gray scale frames are applied at the input of self-organizing map (SOM) described in the next section.

C. Self Organizing Map

Self-Organizing Map (SOM) is a neurocomputational algorithm to map high-dimensional data to a lower dimensional space through a competitive and unsupervised learning process. This algorithm is frequently used to visualize and interpret large high-dimensional data sets. Important SOM features include information compression while trying to preserve topological and metric relationship of the primary data items. The SOM is both a projection method which maps high-dimensional data space into low-dimensional space, and a clustering method so that similar data samples tend to be mapped to nearby neurons. The SOM is widely used as a data mining and visualization method forcomplex data sets. Self-Organizing Maps (SOMs) have broad applications in pattern recognition, speech recognition, engineering system, medical diagnosis and image segmentation. It is of interest to have some order in the activation of a unit in the feedback layer in relation to the activations of its neighboring units [7, 8, 9].

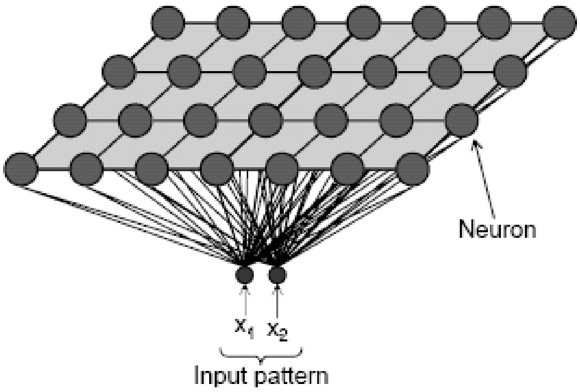


Figure 2: Self Organizing Map

Let us consider the set of input variables $\{x_i\}$ define as the real vector $X=\{x_1, x_2, x_3…x_k\} \in R^n$. This input pattern vector is applied to the processing elements of the input layer. The processing elements of the input layer are connected with each element in the SOM grid (Figure 2). This grid contains the feedback layer region. We associate connection strength W_i = $[\, w_{i_1}, w_{i_2}, \ldots\ldots\ w_{i_n}]^T \in R^n$ to the every processing elements of the feedback layer. The initial value of the W^T is selected randomly.Now the input feature pattern vector X is applied on the processing units of the input layer. The linear output of these processing units fed the weighted input through feed forward connection to the SOM grid. The SOM grid consists with the feedback layer. The activation of the j^{th} process unit of the feed back layer can represent as:

$$y_j = \sum_{i=1}^{K} w_{ij} x_i \qquad \text{-------------(3.2.1.1)}$$

where, j =1 to N (Number of units in the feed back layer)
A winning unit says P is selected among all the processing units of the feedback layer as:

$$\sum_{i=1}^{K}(x_i - w_{P_i}) = \min \sum_{i=1}^{n}(x_i - w_{j_i}) \qquad ;\ \text{for all } j$$

------------ (3.2.1.2)

Hence, during learning the nodes those are topographically close up to certain geometric distance will activate each other to learn from the same input vector X and the weights associated with the winning unit P and its neighboring units r are updated as:

$$w_{iP}(t+1) = w_{iP}(t) + \lambda(P,r)[x_i(t) - w_{ij}(t)]$$

------------- (3.2.1.3)

for i =1 to K and P =1 to N, here the λ (P, r) is the neighborhood function and it can represent as:

$$\lambda(P,r) = \alpha(t).\exp[-\frac{\|R_P - R_r\|^2}{2\sigma^2(t)}]$$

------------- (3.2.1.4)

where R_P refers to the position of the P^{th}unit in the grid, α(t) is learning rate factor (0<α(t) <1) and the parameter σ(t) define the width of the Gaussian function. The σ(t) gradually decreases to reduce the neighborhood region in successive iterations of the training process [10,11].

SOM Algorithm

1. Randomly initialise all weights.
2 .Select input vector x = [x1, x2, x3, … ,xn]
3. Compare x with weights wj for each neuron j to determine winner
4. Update winner so that it becomes more like x, together with the winner's neighbours.

5. Adjust parameters: learning rate & 'neighbourhood function'.
6. Repeat from (2) until the map has converged or pre-defined no. of training cycles have passed.

D. Multimedia Synchronization

The synchronization in multimedia systems refers to the temporal relations between media objects in multimedia systems. In future multimedia systems synchronization may also refer to spatial, content as well as temporal relationships. Synchronization between media objects comprises relationships between time-dependent media objects as well as time-independent media objects [12].Intra-object synchronization refers to the time relation between various presentation units of one time-dependent media object. An example is the time relation between the single frames of a video sequence. For a video with a rate of 25 frames per second, each of the frames must be displayed for 40ms (Figure 3). Inter-object synchronization refers to the synchronization between media objects. Figure 4 shows that the time relations of a multimedia synchronization that starts with an audio/video sequence, followed by several pictures and an animation that is commented by an audio sequence.

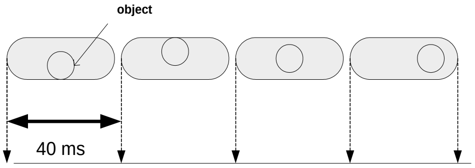


Figure 3: Intra object Synchronization

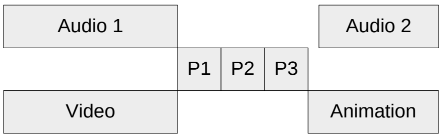


Figure 4: Inter-object Synchronization

A good synchronization algorithm must guarantee both inter-media and intra-media synchronization within the given tolerable precision which is application dependent (Figure 5). At the same time, some video frames (V1,V2…..Vn) the associated audio segment in the same time interval should also be played back precisely under the enforced time-constraint to insure the intermedia synchronization. Some primitive functionality such as multiprocessors/multithreads creation, termination and intercommunication have been assumed for multitasking have been supported by the operating system.

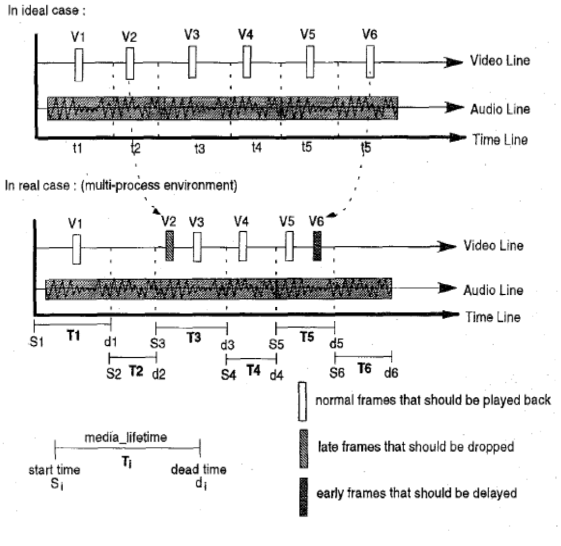


Figure 5: Media lifetime concept and the delay-or-drop policy.

The synchronization can be achieved because the audio buffer is always filled and the audio data in the buffer is consumed at a constant rate, i.e., the audio can be played out smoothly along with the time axis and no audio discontinuity will occur.

According to the proposed model each child media process must undertake both the intra-media and the inter-media synchronization. In order to avoid the audio break phenomenon, the audio-process is designed to keep writing the sequence of audio segments to the audio device in a loop way. On the other hand, video-process adopts a delay-or-drop policy to deal with the out-of synchronization problem, as shown in Table I, based on the definition of playback time interval (the so-called media-lifetime) [13, 14].

Delay-or-drop policy: The corresponding video frame should be: Rule (1): dropped if the current-time is ahead of the dead-time. Rule (2): delayed if the current-time lags behind the start-time. Rule (3): played back if the current-time is within the media lifetime (i.e. in between the start-time and the dead-time) (Table 1).

E. Java Media Framework (JMF) and Real-time Transport Protocol (RTP)

The JMF RTP APIs are designed to work seamlessly with the capture, presentation and processing capabilities of JMF. Players and processors are used to present and manipulate RTP media streams just like any other media content. JMFprovides support for media playback, capturing and storing media data and performing custom processing on media streams. The media streams are then transmitted over the network where all the nodes are connected in a unicast or multicast network [15, 16].

TABLE I

Table showing the Pseudo codes of the video and audio process

Video Process	Audio Process

1. loop{
2. retrieving_video_frame(i);
3. get current_time from system clock;
4. if(current_time>dead_time((i)){
5. drop this video frame i;
6. jump to next appropriate frame i;
7. }
8. if(current_time<start_time((i)){
9. delay until(current_time>= start_time((i));
10. }
11. playback_video_frame(i);
12. }until end_of_playback
	1.loop{
	2.retrieving_audio_segment(i);
	3.playback_audio _segment(i);
	4.}until end_of_ playback

The Real-Time Transport Protocol (RTP) handles transport issues specifically related to real-time data. RTP includes another protocol, Real-Time Control Protocol (RTCP), for managing RTP sessions. Some of the main responsibilities of RTP/RTCP are: packet sequencing, synchronization, payload identification, QOS feedback and encryption. Real-time Streaming Protocol (RTSP) is a protocol for initiating and controlling the delivery of both stored and live multimedia streams over the Internet. The main concept of RTSP is providing a session-like abstraction for delivering one or more media streams to a single client or multicast destination [19, 20].

IV. Design of System

A. *Video compression using SOM*

The function of the entire system is described by the the following block diagram (Figure 6). The video sequence is given as an input to the system. The video frames are extracted from the video sequence that are stored in memory in gray scale format for further processing. The number of extracted video frames depends on the frame rate and all video frames are of equal size (120X160). Here we are using the frames rate as 15 frames per second (fps). The frames are then passed to the self-organizing map (SOM) that outputs the useful features from the video frames. The grid size of the SOM depends on the number of neurons selected during the initialization of the SOM.

The features obtained from SOM are stored in the Feature Vector (FV=FV1, FV2……..FV15) and it is used for encoding the video frames. In this paper if we are using 100 neurons then the SOM grid size is 10X10 and the feature vector size is 100x1 for eg. FV1=100X1, FV2=100X1…………FV15=100X1.

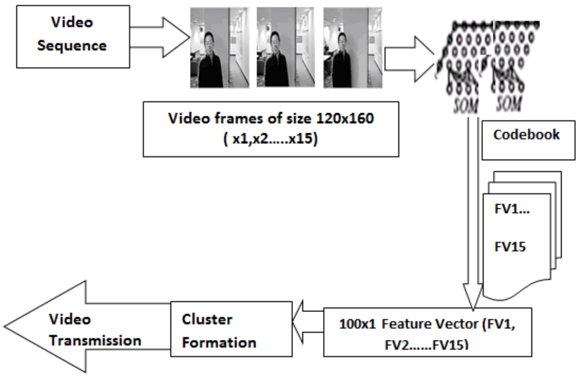

Video Sequence
Video frames of size 120x160 (x1,x2.....x15)
SOM
SOM
Codebook
FV1...
FV15
100x1 Feature Vector (FV1, FV2......FV15)
Cluster Formation
Video Transmission

Figure 6: Block diagram of the system

The feature vector obtained are then passed through the cluster formation step in which the similar feature vectors are grouped together and then form clusters. The codewords generated using SOM are used in encoding and decoding of video frames, and are also used for video frames reconstruction.

Video transmission over network using JMF

A typical video streaming system is shown in Figure 7 and it works as follows: the system is composed of a source and a receiver, or a group of receivers, which all become members of a multicast group by joining this group using a multicast IP address and a given port number. The source sends to the receiver its compressed stored video or live video content for the streamed session on a RTP stream. The receiver is then capable of receiving the RTP stream and playing them back.

During the session time each receiver issues a series of RTCP reports for each received stream periodically. These reports are destined to the same multicast IP address and port number. They help in identifying the most recent receiving status of the receiver mainly regarding jitter, the number of packets lost and its fraction from the sent packets.

The session manager presence is essential to avoid source overloading or crashing if the monitoring process is also left to the source to handle. The session manager's logically exists between the source and the receiver and also joins the same multicast group.

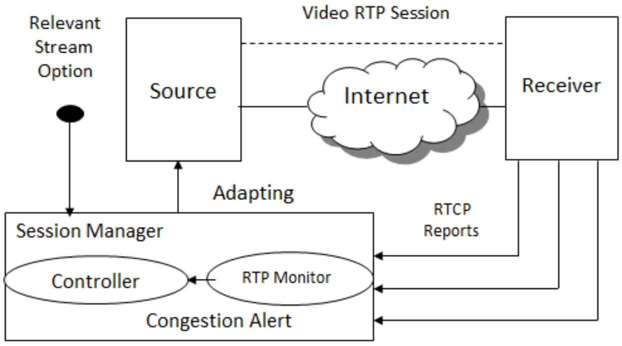
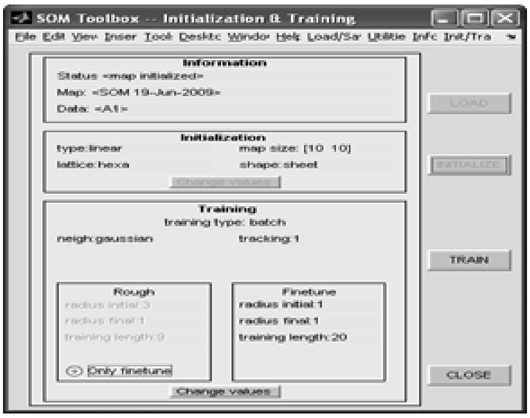


Figure 7: Video transmission System

V. Results & Discussion

The results presented in this section have demonstrated that the video frames are extracted from the video sequence by preprocessing the video. The video frames are then loaded to the following SOM Graphical User Interface (GUI) for extracting the features from all the video frames (Figure 8). The redundant information is removed from the video frames and dimension of the frames are reduced by encoding all the information of video frames into feature vectors. The SOM allows us to see similar entities placed in the same map unit or adjacent map units.


SOM Toolbox -- Initialization & Training
File Edit View Inser Tool Desktc Windo Help Load/Sa Utilitie Infc Init/Tra
LOAD
INITIALIZE
TRAIN
CLOSE
Change values
Only finetune
Change values

Information
Status <map initialized>
Map: <SOM 19-Jun-2009>
Data: <A1>

Initialization
type:linear
map size: [10 10]
lattice:hexa
shape:sheet

Training
training type: batch
neigh:gaussian
tracking:1

Rough
radius initial:3
radius final:1
training length:9

Finetune
radius initial:1
radius final:1
training length:20

Figure 8: Initialization GUI

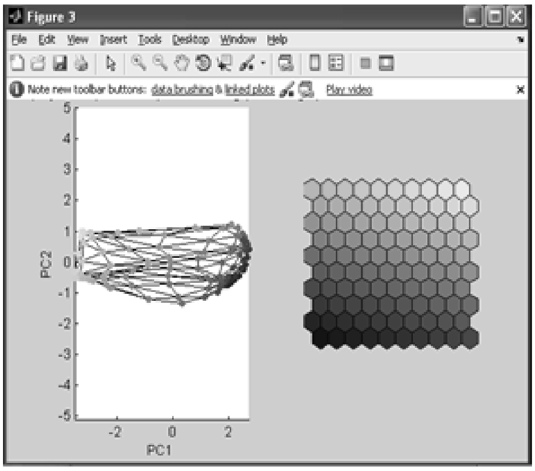


Figure 3
File Edit View Insert Tools Desktop Window Help
Note new toolbar buttons: data brushing & linked plots
Play video
5
4
3
2
1
0
-1
-2
-3
-4
-5
PC2
-2
0
2
PC1

Figure 9: SOM of first video frame

The map size feature of SOM is used to initialize the SOM grid size and number of neurons. According to the SOM grid size feature vector is obtained. Here we used 10x10 SOM grid size or 100 neurons for each video frames. SOM map obtained after training the map is shown in Figure 9. In this case, 100X1 feature vector is obtained for each video frame.

The codeword and feature vector of 10 X10 SOM grid or 100 neurons is shown in Table II. If the frame rate is 15 frames per second then 15 Feature Vectors (FV_1,FV_2……FV_{15}) are obtained. These feature vectors are obtained from the codewords generated from SOM. If the codeword (value) > 0.5 then FV(value)=1 else FV(value)=0. These feature vectors are clustered and then used for the transmission over the network using the Java Media Framework.

Thus the video frames each of size 120 X 160 is compressed to 100 X 1 feature vector.The feature vectors are compared and common feature vectors are clustered into groups.

TABLE II
Table showing codewords generated from SOM and feature vector in 100 X1 vector

Code word	Feature Vector (FV1)	Code word	Feature Vector (FV2)	..	Code word	Feature Vector (FV(15)
0.38066	0	0.38724	0	..	0.40818	0
0.36457	0	0.34986	0	..	0.39175	0
0.32925	0	0.4203	0	..	0.37674	0
0.62242	1	0.68367	1	..	0.38136	0
0.6305	1	0.64707	1	..	0.68683	1
. . . .				..	. .	
. . . .				..	. .	
100 X 1	100 X 1	100 X 1	100 X 1		100 X 1	100 X 1

For eg: if FV1=FV2 then FV1 and FV2 are clustered into one group.That cluster is used for the further video processing by transmitting the compressed video over the network.

The audio and video data is transmitted in unicast and multicast network. It is first captured, then transmitted through network and then received on the receiving side. The synchronization of the streams is done by using buffering techniques in case of different media streams and then played on the receiving side. The same method is applied on real time media streams by capturing streams and use Real-time Transport Protocol (RTP). At the receiving side the video is received by setting the IP address and port number of the sending side (Figure 10).

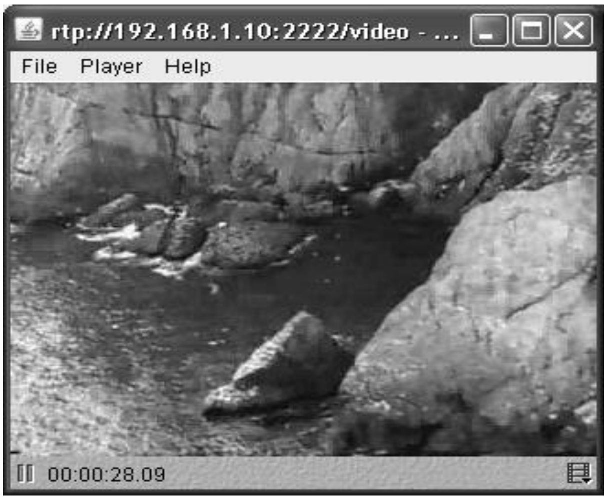


Figure 10: Reception of media streams on receiving side

VI. Conclusion & Future Plan

We used SOM for feature extraction from video frames that are obtained by preprocessing the video sequence. The codewords are obtained corresponding to each video frames and are converted into the feature vector. The feature vectors obtained from SOM are in compressed form. The feature vectors are grouped together and clusters are formed that are used for the transmission over the network in compressed form using Java Media Framework and Real-time Transport Protocol. This work will be extended for transmission over the overlay networks.

VII. References

[1]A. K. Jain (1989). Fundamental of Digital Image Processing, Prestice-Hall, Englewood Cliffs, NJ.

[2] W. K. Pratt (2007). Digital Image Processing, 3rd ed., Wiley-Interscience, New York.

[3] Rafael C. Gonzalez, Richard E. Woods, Steven L.Eddins (2003). Digital Image Processing Using Matlab.Pearson Prentice Hall.

[4] T. Kohonen (1990). "The Self-Organizing Map," Proceedings of the IEEE, vol. 78, no. 9, pp. 1464-1480.

[5] C.Amerijckx, M. Velerysen, P. Thissen& J.D. Legat (1998). "Image compression by self organized Kohonen map", IEEE Transactions on Neural Networks, 9 (3), 503-507.

[6] Chen, O.T.-C., B.J. Sheu & W.-C. Fang (1994). "Image compression using self organization networks", IEEE Trans. Circuits Syst. Video Tech., 4 (5), 480-489.

[7] H. S. Kong, L. Guan & S. Y. Kung (2002). "A self organizing tree map approach for image segmentation", Proceedings of ICSP, 588-591.

[8] Y. J. Zheng (1994). "Self organizing grouping for feature extraction and image segmentation.International symposium on speech", Image Processing and Neural Networks, 13-16, April.

[9] Retrieved on 2006-06-18. "Intro to SOM by TeuvoKohonen", SOM Toolbox.

[10] N. Nasrabadi and Y. Feng (1988). "Vector quantization of images based upon the Kohonen Self-organizing Feature Maps," in IEEE Int. Conf. Neural Networks, San Diego, CA, vol. 1, pp. 101–108.

[11] S. Kotsiantis, P. Pintelas (2004). Recent Advances in Clustering: A Brief Survey, WSEAS Transactions on Information Science and Applications, Vol 1, No 1 (73-81).

[12] G. Blakowski and R. Steinmetz; "A Media Synchronization Survey: Reference Model, Specification, and Case Studies," IEEE J. Selected Areas in Comm., vol. 14, no. 1, Jan. 1996, pp. 5-35.

[13] Little, T. D. C; Ghafoor;"A. Synchronization and Storage Models for Multimedia Objects", IEEE Journal on Selected Areas in Communications, Apr., 1990.

[14] C.Huang, H.Kung, and J.Yang; "Synchronization and flow adaptation schemes for reliable multiple-stream transmission in multimedia presentations", Journal of Systems and Software, V. 56, Issue 2, 1 March 2001, Pages 133-151.

[15]JMF 2.0 Programmer's guide; http://java.sun.com /products/java-media/jmf/2.0/jmf20-08-guide.pdf"

[16] L. Berc, W. Fenner, R. Frederick, S. McCanne, and P. Stewart;"RTP payload format for JPEG-compressed video," Internet Engineering Task Force, RFC 2435, 1998.