Using Unitary Matrices in High-Speed Video Encryption

Hala B. Abdul Wahab† and May A. Salih††,

University of Technology , University of Babylon

Summary
Encryption computational complexity and encryption speed are two important aspects of video encryption algorithms. Due to its increasingly large size, video images present a great challenge to currently available cryptographic algorithms; the processes of encryption and decryption of video images are so computationally intensive that they introduce delays beyond acceptable real-time application limits. [1] In this paper we introduce unitary matrices as both encryption keys and the control stream to verify which key will be used for each block. Showing how it is more efficient than the traditional Gauss Jordan elimination from time perspective especially for applications that are time dependent. The study case showed in this paper works on GF (p) and for encryption key sizes varying from 3X3 to 12X12 .The goal is to provide a highly secure encryption algorithm with a wide space for encryption speed.

Key words:
High-speed, video encryption, video-on-demand, unitary matrices, Gauss Jordan

1. Introduction

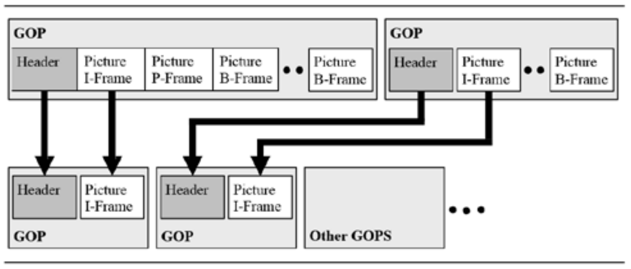
Not all Streaming Server platforms implement support for high-speed playback. Support for these playback modes is complex and uses valuable resources on the server. Similarly, the most common use of high-speed playback modes is to locate a particular scene, this functionality is already provided with indexed playback (if the scene timestamp is known). For the server platforms that do provide this high-speed playback, it is not implemented by streaming the installed bit-stream over the network at a higher bit-rate. This increases both the disk and bandwidth requirements for an individual stream, and will result in a lower number of concurrent streams that can be supported by the server. Bandwidth requirements can be minimized in two ways, one is to not stream the encapsulated Audio Stream as there is no need to perform Audio Playback in a high-speed mode. The second technique is to not stream every Video Frame – as playback occurs in high-speed, it is not necessary to display all frames. If this approach is carefully undertaken, it is possible to stream an MPEG bit-stream in a high-speed playback mode with similar bandwidth requirements to streaming at normal playback speed [2]. During the development of network and multimedia technology, more and more images transmit over the Internet and through the wireless networks. Digital images have become one of the most important information carriers, which is helpful for people to communicate with each other. However, because of the intrinsic features of video images, such as bulk data capacity and high correlation among pixels, it is not suitable for practical image encryption, especially during on-line communications. Therefore, people begin to explore dynamic Galois field matrices which is more efficient at hiding image information[3]. Image encryption has applications in internet communication, multimedia systems, medical imaging, telemedicine, military communication, etc. Images differ from text in that they are bigger in size and the decrypted image must be equal to the encrypted image. Another significant point to consider in transfer of digital images is their special attributes like bulk data capacity, high redundancy, and high correlation between neighboring pixels [4][5].

2. High-speed MPEG Video Stream

High-speed streaming is generally performed by streaming a modified MPEG Video Stream (containing only I-Frames). As this stream must be decoded at the client end, this newly created bit-stream must be a valid MPEG Video Stream. A properly formatted Video Stream can be constructed using the following steps:
•Existing Group of Pictures (GOP) Headers do not specify the number of frames contained within the GOP. As such, it is possible to remove any number of frames within the GOP without changing the contents of the GOP Header itself.
•Existing Picture Headers specify only the frame number in presentation order within the GOP. Since the I-Frame always forms the first frame of the GOP its frame number will be 1. As such, it is possible to remove other Pictures from the GOP without changing the contents of the Headers of the remaining Pictures.
•An existing Video Stream is still valid if individual Pictures are removed from it. By removing all bar the first Picture within each GOP, the result is a series of GOPs, each containing a solitary I-Frame.
•When streaming this new bit-stream in reverse mode, each GOP should be transmitted in reverse order, as each GOP contains one Picture, each I-Frame will be decoded and displayed in reverse order – simulating the effect of a reverse, high-speed playback mode.
•Indexed High-Speed playback can be achieved by commencing the stream at the start of any of the newly

Manuscript received August 5, 2014
Manuscript revised August 20, 2014

created GOPs. This bit-stream needs to be pre-pended by a copy of the Video Stream Sequence Header to create a valid MPEG Video Bit-stream [2].



Figure(1) Modified MPEG-1 Video Sequence to Implement High-Speed

3. Gauss Jordan Elimination Method

Traditional way to compute encryption and decryption keys is by using the usual notation, by C_r^(m×n) we denote the set of all complex m×n matrices of rank r, and by I we denote the unit matrix of an appropriate order . Furthermore A*, R(A), rank (A) and N(A)denote the conjugate transpose, the range, the rank and null space of A∈C_^(m×n).

If A ∈ C_r^(m×n). This a subspace of C_^n of dimension t ≤ r and S is subspace of C_^m of dimension m-t, then A has a {2}- inverse X such that R(X)=T and N(X)=S if and only if AT□S=C_^m. In the case when the existence is ensured, X is unique and it is denoted by A_(T.S)^((2)).We study Gauss Jordan elimination methods for computing various inverses of square matrices. The oldest and best known among these methods is the method for calculating the inverse matrix. The Gauss Jordan elimination method for computing the inverse of a nonsingular matrix A is based on the executing elementary row operations on the pair [A | I] and its transformation into the 7 block matrix involving the inverse A-1. A number of numerical methods are developed for computing various classes of outer inverses with prescribed range and null space. The Gauss Jordan elimination method to compute the Moore Penrose inverse is developed in [6]. The method from [6] is based on two successive sets of elementary row operations.

4. Unitary and Hermitian Matrices

Problems involving diagonalization of complex matrices, and the associated eigenvalue problems, require the concept of unitary matrices. These matrices roughly correspond to orthogonal and symmetric real matrices. In order to define unitary matrices, we first introduce the concept of the conjugate transpose of a complex matrix.
Definition 1: The conjugate transpose of a complex matrix A, denoted by A*, is given by

A*=A-T where the entries of A-T are the complex conjugates of the corresponding entries of A.
Unitary Matrices: Recall that a real matrix A is orthogonal if and only if A*=A-TIn the complex system, matrices having the property that A-1=A* are more useful and we call such matrices unitary.
Example 1 Show that the following matrix is unitary.

$$A = \frac{1}{2}\begin{bmatrix} 1+i & 1-i \\ 1-i & 1+i \end{bmatrix}$$

Since $AA^* = \frac{1}{2}\begin{bmatrix} 1+i & 1-i \\ 1-i & 1+i \end{bmatrix}\frac{1}{2}\begin{bmatrix} 1+i & 1-i \\ 1-i & 1+i \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix} = I_2$

We conclude that A*=A-T. Therefore, A is a unitary matrix [7].
Let w be a complex vector. Define the elementary Hermitian matrix U = I -2wwT where wTw = 1. It is easily verified that U is both Hermitian and unitary. In particular, if w is a real vector, then U is orthogonal and symmetric, and is commonly referred to as a Householder reflector. Since U is unitary, its inverse is readily available.

5. The proposed Video Encryption algorithm

The proposed schema is based on two types of keys: encryption key and control key. The control key is generated using a controlled ranged randomized stream and the encryption keys are a set of non-fixed variable (GF(P)) unitary matrices
The first step is to detect video data from video file. The next step multiplies variable-sized blocks with the opponent key matrix based on the control key as showed in fig (2).

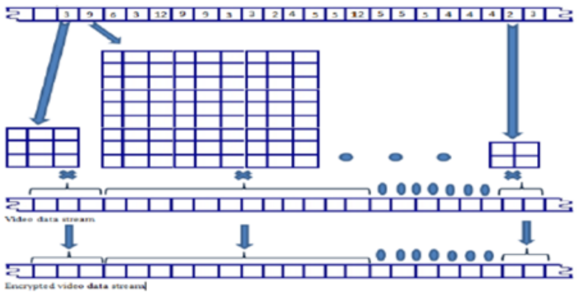
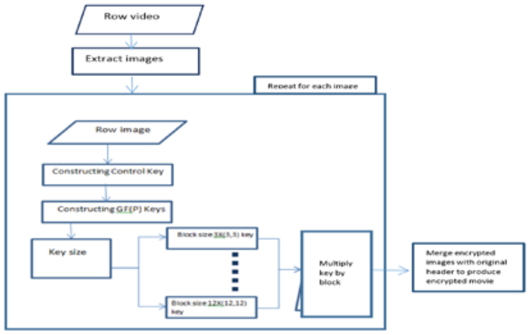


Fig (2) Proposed Encryption algorithm

Encryption in digital video images mostly works at pixel level, which is the lowest level of information in the image. But because of strong correlation between neighboring pixels, one can easily decode data for one pixel if that of a neighboring pixel becomes known. However, an image can also be interpreted as an ordered arrangement of image blocks instead of pixels. An accurate orientation of these image blocks lets us infer information from the image, where any change causes visual disruption [5]. The proposed algorithm is shown in fig (2). Thus, using block level encryption for images will help overcome the chief nuisance in image encryption, correlation between neighboring pixels. The block size should be smaller for better transformation because then fewer pixels will keep their neighbor's data [8]. These advantages of blocks over pixels are why block encryption is preferred over stream cipher. However, a considerable drawback of block cipher is that it produces the same cipher text for the same plaintext if it is encoded with the same key. In response, we propose an image encryption technique of a partial symmetric-key algorithm.it is not fully dependent on the secret key and hence achieves better computational security against unauthorized attacks. It actually uses two keys for encoding; one at the block level and other at the pixel level.

6. The proposed Video Encryption steps.

Step 1. Generate Control Key
Step 2. Generate GF(p) masks and inverses for sizes [3-12] using unitary matrices
Step 3. Prepare video data
Step 4. Repeat until end of file
A .Skip Header and detect video data
B. repeat until end of video data
1. Pick control key value
2. Select mask dimension = control key value
3. Determine size of block = control key value
4. Apply function like encrypted block = mask * block
5. Control key controller +1
    6. Check for end of video data
  7. Control key controller +1
  8. If header value appeared then go to A
  7. Check or end of file
Step 5. Collect encrypted blocks to form encrypted stream with keeping the same header information
Step 6. Merge encrypted stream with original header.
Step 7. End
Fig(3) the proposed algoritm flow diagram



Fig(3) the proposed algoritm flow diagram

7. Computational Time for Finding the Inverse of a Matrix.

The execution time of a program depends on the number of *floating*-point operations (FLOPs) involved. Every computer has a processor speed which can be defined in flops/sec. knowing the processor speed and how many flops are needed to run a program gives us the computational time required:
Time required (sec) = Number of FLOPs/Processor Speed (FLOP/sec)
A supercomputer may be capable of 50 $x10_{12}$ FLOPs per second, while a typical PC may be capable of 10 $x10_{9}$ FLOPs per second [9].

8. Time to compute Inverse using Gaussian Elimination [9].

To find the inverse of a *nxn* matrix, one can use Naïve Gaussian Elimination method. For calculations of *n* columns of the inverse of the matrix, the forward elimination and back substitution needs to be done *n* times. Complete details of Naïve Gauss Elimination are given here.
The following formulas define the number of FLOPs for each step of Naïve Gauss method.
Forward Elimination (FENG): The FLOPs used in the forward elimination step of Naïve Gauss for a set of n equations is given by the series

$$\sum_{k=1}^{n-1}(n*(n+2)-k*(2*n+2)+k^2)$$

$$\frac{1}{6}((-1+n)n(5+2n))$$

When expanded, the number of FLOPs used is equal to

$$FENG = Expand\left[\sum_{k=1}^{n-1}(n*(n+2)-k*(2*n+2)+k^2)\right.$$

$$-\frac{5n}{6}+\frac{n^2}{2}+\frac{n^3}{3}$$

Back Substitution (BSNG): The FLOPs used in the back substitution step for a set of n equations is given by the series

$$\sum_{i=1}^{n} i$$

$$\frac{1}{2}\,n\,(1+n)$$

When expanded, the number of FLOPs is equal to

$$BSNG = Expand\left[\sum_{i=1}^{n} i\right]$$

$$\frac{n}{2}+\frac{n^2}{2}$$

Total number of FLOPs required to find the inverse of the [A] matrix using Naïve Gaussian Elimination is n*(FE+BS)which is equivalent to:

NGFLOP = Expand [n ∗(FENG + BSNG)]

$$-\frac{n^2}{3}+n^3+\frac{n^4}{3}$$

9. Unitary matrices Vs. Gauss Jordan Elimination

For a small square matrix, let us say n=10, the number of floating-point operations using Naïve Gaussian elimination is:

NGFLOP /. n → 10 is 4300

For the same size matrix, the number of FLOPs using unitary matrices is

LUFLOP /. n → 10 is 1143

10. Conclusion

For a matrix of this size, Naïve Gaussian method requires nearly 3 (or approximately n/4) times more FLOPs than Unitary elimination. However, if one were to calculate the FLOPs required for a square matrix with an order of 100,one can see that, although the order of the matrix increases 10 fold, the number of FLOPs for Naïve Gaussian Elimination requires nearly 20 (or approximately n/4) times more FLOPs than Unitary elimination.

FLOPs for Naïve Gauss:

NGFLOP /. n → 100 is 34330000

FLOPs for Unitary elimination:

LUFLOP /. n → 100 is 114220

Table (1). Shows how long it would take to encrypt multiple files

File type	Gauss Jordan	Unitary elimination
256*256 image	3.01 Sec.	1.54 Sec.
512*512 image	4.8 Sec.	2.04 Sec.
300*300 MPEG video Length 2 Min.	42.1 Sec.	17.8 Sec.
600*600 MPEG video Length 2 Min.	1.34 Min.	34.06 Sec.

While some researchers compute only encryption/ decryption time but what was included in this research is the whole process time that is key and inverse of key construction time plus the encryption / decryption process

File type	Gauss Jordan	Unitary elimination
256*256 image	1.4 Min.	1.54 Sec.
512*512 image	3.2 Min	2.04 Sec.
300*300 MPEG video Length 2 Min.	8.1 Min	17.8 Sec.
600*600 MPEG video Length 2 Min.	14 Min.	34.06 Sec.

As we found that Unitary Elimination is faster in encryption about 3-4 times than Gauss Jordan and Unitary Elimination is faster in decryption about 14 times than Gauss Jordan.

References

[1] "Performance Study of a Selective Encryption Scheme for the Security of Networked, Real-Time Video "Las Vegas, Nevada, USA September 20-September 23 ISBN: 0-8186-7180-7

[2] Jason But "A Novel MPEG-1 Partial Encryption Scheme for the Purposes of Streaming Video" Doctor of Philosophy thesis, Department of Electrical and Computer Systems Engineering Monish University, January 2004

[3] "A Symmetric Image Encryption Scheme Based on Composite Chaotic Dispersed Dynamics System" Zhenzhen Lv1, Lei Zhang2, and Jiansheng Guo 3,Proceedings of the Second Symposium International Computer Science and Computational Technology(ISCSCT '09) Huangshan, P. R. China, 26-28,Dec. 2009, pp. 191-194

[4] Zhang S. and M. A. Karim, pp. 318-322, Vol. 21, No. 5, June 5 1999, Color image encryption using double random phase encoding, Microwave and optical technology letters.

[5] Maniccam S.S and Bourbakis N.G., pp. 1229-1245, 2001, Lossless image compression and encryption using SCAN, Pattern Recognition

[6] Predrag S. Stanimirovi_c1, Marko D. Petkovi_c2,"Gauss Jordan elimination method for computing outer inverses " .

[7] Ron Larson, Bruce H. Edwards, David C. Falvo," Elementary Linear Algebra", Houghton Mifflin Harcourt Publishing Company, 2003

[8] Quist-Aphetsi Kester, "A cryptographic Image Encryption technique based on the RGB Pixel shuffling," International Journal of Advanced Research in ComputerEngineering & Technology (IJARCET), vol. 2, issue 2, January 2013, pp. 848-854

[9] Jamie Trahan, Autar Kaw, Kevin Martin,"Computational Time for Finding theInverse of a Matrix:LU Decomposition vs. Naive Gaussian Elimination "nbm sle sim invers-ecomptime.nb