

Video Compression using Wavelets

A dissertation submitted in partial fulfillment
of the requirements of M.Tech.(Computer Science)
degree of Indian Statistical Institute, Kolkata

by

Yellajosyula Sailaja

under the supervision of

Dr. C. A. Murthy
Machine Intelligence Unit

Indian Statistical Institute
203, Barrackpor Trunk Road
Kolkata-700 108.

July 22, 2002


Indian Statistical Institute

203, Barrackpore Trunk Road,

Kolkata-700 108.

Certificate of Approval

This is to certify that this thesis titled "Video Compression using Wavelets" submitted by **Yellajosyula Sailaja** towards partial fulfillment of requirements for the degree of M. Tech in Computer Science at Indian Statistical Institute, Kolkata embodies the work done under my supervision.

 17.7.02.

C. A. Murthy,
Machine Intelligence Unit,
Indian Statistical Institute,
Kolkata-700 108.

Acknowledgements

I am introduced to the field of image processing in general and video compression in particular by Prof. C. A. Murthy. I am thankful to him for his constant guidance, encouragement and criticism, during all stages of project work, which have benefitted me immensely in giving the dissertation its present shape.

I am thankful to Prof. M. K. Kundu and Prof. Tinku Acharya for their useful discussions and assistance during the earlier phase of my work. Finally I express my thanks to Head, Machine Intelligence Unit, Indian Statistical Institute for providing access to the department's resources.

Yellajosyula Sailaja

Contents

1	Introduction	5
1.1	<i>Why Compression</i>	5
1.2	<i>Applications of Video Compression</i>	5
1.3	<i>Relevant methodology</i>	6
1.4	<i>Work carried out in this project</i>	9
1.5	<i>Organisation of thesis</i>	10
2	Wavelets for Image Compression	11
2.1	<i>What are wavelets</i>	11
2.1.1	<i>Mother wavelet</i>	11
2.1.2	<i>Wavelet transform</i>	12
2.1.3	<i>Father wavelet</i>	14
2.1.4	<i>Multi resolution analysis</i>	14
2.1.5	<i>How to construct Wavelets</i>	14
2.2	<i>Examples of Wavelets</i>	15
2.3	<i>Wavelets chosen for this project</i>	16
2.4	<i>Why wavelets for Compression</i>	16
2.5	<i>Image Compression</i>	17
2.6	<i>Image Compression using wavelets</i>	17
2.7	<i>Still Image Compression techniques using wavelets - JPEG 2000</i>	21
3	Video Compression	22
3.1	<i>Video Compression - an Introduction</i>	22
3.2	<i>Methods for Video Compression</i>	22
3.3	<i>What is a motion vector</i>	23
3.4	<i>Video Compression Standards - using motion vector</i>	23
3.4.1	<i>The H.261 Standard</i>	23
3.4.2	<i>The MPEG Standard</i>	26
3.5	<i>Why a new method</i>	28

4	Proposed wavelet based Video Compression method for any video	29
4.1	<i>Algorithm</i>	29
4.1.1	<i>Intra frame coding</i>	29
4.1.2	<i>Inter frame coding</i>	30
4.2	<i>Data sets</i>	32
5	Proposed method for Video Compression using Macro motion vector	33
5.1	<i>What is a Macro motion vector</i>	33
5.2	<i>Algorithm</i>	34
5.2.1	<i>Intra frame coding</i>	34
5.2.2	<i>Inter frame coding</i>	35
5.3	<i>Data sets</i>	36
6	Implementation details	37
6.1	<i>Discrete Wavelet Transform(DWT)</i>	37
6.2	<i>Quantization</i>	38
6.3	<i>Modified Huffman and run length coding</i>	38
6.4	<i>Finding a motion vector</i>	39
6.5	<i>Finding a Macro motion vector (MMV)</i>	41
6.5.1	<i>Finding a best block</i>	41
6.6	<i>Video Compression - Implementation</i>	43
7	Results and Comparisons	44
7.0.1	Results of First method	44
7.0.2	Results of Second method	48
7.0.3	<i>Comparisons</i>	54
8	Conclusions, discussion and scope for further work	55
8.1	<i>Conclusions</i>	55
8.2	<i>Discussion and scope for further work</i>	55

Chapter 1

Introduction

1.1 *Why Compression*

A video stream consists of a series of still images or frames displayed in rapid succession. Transmitting video over a network can be expensive in terms of bandwidth utilization. When more bandwidth is consumed by video, less is available for other applications such as voice, data and mission critical systems. In addition to this, over wide area networks where usage charges may apply, transmitting large video streams can become cost prohibitive. Limited storage online has fed the need to compact this information into the smallest size possible. The development of compression programs such as "zip" have sufficiently met this need for data, but leave streaming media such as video and audio untouched. As the trend to display video on the desktop grows, it is vital that this data gets compressed. Digital video could only be used in wider applications if the storage and bandwidth requirements could be eased; easing these requirements is the purpose of compression. In this project two specific compression techniques for video frames are addressed.

1.2 *Applications of Video Compression*

Video compression is used in many applications to reduce the amount of information required to represent a sequence of frames. There is a tremendous variety of applications for video compression. Some of them are Video Conferencing, Video phone, Video mail, Video games, Video Library, Broadcasting, Video-on-demand, Cable TV distribution.

1.3 *Relevant methodology*

Video Compression techniques are becoming increasingly important due to present and forthcoming visual services and applications on the internet. This ever-expanding area has generated a lot of research and increased standardization activities.

Some of the methods available on video compression are as follows.

- ***An elementary approach*** [1]

An elementary approach to video compression would be to employ any of the still image compression techniques on a frame by frame basis. However, the compression that can be achieved by such an approach will be limited because each frame is treated as an independent image.

The methods which are later developed are *interframe* compression methods which exploit the temporal redundancies, due to similarity between neighbouring frames, to provide superior compression efficiency.

- ***Subsampling*** [2]

The encoder selects every alternate frame and writes it on the compressed stream. This yields a compression factor of 2. The decoder inputs a frame and duplicates it to create two frames.

- ***Differencing*** [2]

A frame is compared to its predecessor. If the difference between them is small, the encoder encodes the pixels that are different by writing three numbers on the compressed stream for each pixel : its image coordinates, and the difference between the values of the pixel in the two frames. If the difference between the frames is large, the current frame is written on the output in the raw format.

A lossy version of differencing looks at the amount of change in the pixel. If the difference between the intensities of a pixel in the preceding frame and in the current frame is smaller than a certain threshold the pixel is not considered different.

- ***Block Differencing*** [2]

Each block B in the current frame is compared to the corresponding block P in the preceding frame. If the blocks differ by more than a certain amount then B is compressed by writing its image coordinates followed by the values of all its pixels on the compressed stream.

- ***3-D Waveform Coding*** [1]

A simple way to extend still-frame image compression methods to interframe video compression is to consider 3-D waveform coding schemes, which include 3-D transform and 3-D

subband coding. These methods exploit spatio-temporal redundancies in a video source through statistical signal models.

3-D Transform Coding [1]

3-D DCT coding is a straightforward extension of the 2-D DCT coding method. In this method the video is divided into $M \times N \times J$ blocks, where M is the number of rows in a frame, N is the number of columns in a frame and J is the number of video frames.

The transform coefficients are then quantized and encoded. A common problem with this method is blocking artifacts due to which it has not been widely used in practical applications.

3-D Subband Coding [1]

3-D subband coding is an extension of 2-D subband coding. In this, the video is decomposed into various properly subsampled component video signals. These are encoded independently using algorithms adapted to the statistical and psychovisual properties of the respective spatio-temporal frequency bands. Compression is achieved by appropriate quantization of the various components and entropy coding of the quantized values.

This method has received increased attention due to the following reasons:

1. It is almost always free from blocking artifacts.
2. It does not require a separate motion estimation stage.

- ***Motion-Detection based approach [1]***

One of the earliest approaches in interframe image compression is based on segmenting each frame into changed and unchanged regions with respect to the previous frame. The information about the addresses and intensities of the pixels in the changed region would be transmitted using a bit rate that is matched to the channel rate. This technique is called *conditional replenishment* technique. It is a motion-detected based algorithm since it does not require explicit estimation of the motion vectors.

- ***Motion-Compensated Waveform Coding [1]***

Observe that the difference between consecutive frames is small because it is the result of moving the scene, the camera or both between the frames. The knowledge about the movement of camera or objects can be used to get better compression. Motion compensation method is effective if objects are just translated not scaled or rotated. Most commonly used motion estimation methods in motion-compensated (MC) compression are block-matching algorithms.

Two such methods are described as follows.

1. **MC Transform Coding [1]**

In MC transform coding, the displaced frame difference is segmented into blocks and DCT coefficients of each block are encoded as in 2-D DCT coding. The temporal prediction aims at

minimizing the temporal redundancy and the DCT encoding makes use of the spatial redundancy in the prediction error. The MC transform coding is the basis of several world standards for video compression.

Some of the standards are

- a. The H.261 Standard [1].
- b. The MPEG Standard [1].

The details of the MC transform coding will be covered later where we discuss about these world standards.

2. MC Vector Quantization (MC-VQ) [1]

In this method the differential signal is encoded by vector quantization. A typical MC-VQ scheme partitions pixels in the current frame into 4x4 or 8x8 blocks. First, a motion-detection test is applied to each block. The outcome of the motion-detection test determines one of the three options.

1. Do nothing if no motion is detected.
 2. Do interframe VQ if motion is detected and the motion vector can be estimated with sufficient accuracy. The estimated motion vector, one for each block, and the displaced block difference are encoded and then transmitted.
 3. Do intraframe VQ if motion is detected but cannot be estimated within an acceptable accuracy limit. In this case the actual pixel intensities are VQ encoded.
- **Model-Based Coding [1]**

3-D and MC waveform coding provide satisfactory results with images at high bit rates. However, the quality of images that these techniques offer at very low bitrates, like videophone, is deemed unacceptable and the images obtained by these methods generally suffer from blocking artifacts. To this effect, a variety of new motion-compensated coding schemes have recently been proposed for very-low-bitrate applications. These are generally known as model based coders or analysis-synthesis coders.

An analysis-synthesis encoder can be characterized by the following steps :

Image analysis:

The present frame is segmented into individually moving objects using the knowledge of the previously coded frames. Each object in the present frame is characterized by a set of shape and motion parameters.

Image synthesis:

The present frame is synthesized based on the estimated shape and motion parameters and the knowledge of the previously coded frames. The difference between the actual and the synthesized frame provides a measure of model compliance. Those regions where this difference is more than a threshold are labeled as model failure regions.

Coding:

The shape, motion and color (for the model failure regions) parameters are separately entropy encoded and transmitted.

• **Knowledge-Based Coding [1]**

Knowledge-Based Coding is a model based coding scheme. It deals with cases where we have some a priori information about the content of the video. For example, in videophone applications, head-and-shoulders-type images are common. Here source model is a moving known object characterized by the following.

1. A generic wire frame model to describe the shape of the object.
2. 3-D global motion parameters.
3. 3-D local motion parameters.
4. Color parameters.

Coding algorithm :

1. Detect the boundaries of the object in the initial frame.
2. Adapt the generic wireframe model to the particular object.
3. Estimate the 3-D global and local motion parameters.
4. Synthesise the next frame.
5. Determine the model failure areas.
6. Code the motion and color parameters.

1.4 *Work carried out in this project*

In general there are three types of video sequences available. They are described below.

1. Type 1 sequence : Here camera is kept fixed and the objects are moving.
2. Type 2 sequenec : Here camera is kept moving but the objects are fixed.

3. Type 3 sequence : Here none of the camera and the objects are fixed to a location.

The methods developed in this project for video compression are

- Video Compression without using motion vectors.
- Video Compression using motion vectors.

First method works for all the 3 types of video. It yields good quality with less compression whereas second method is restricted to video of type 2 i.e camera is kept moving but the objects are fixed and it gives a good compression ratio. This method makes use of motion vectors. We will discuss about these methods in Chapters 4 and 5.

The video images under consideration are gray level square images. We are not dealing with color images here.

1.5 *Organisation of thesis*

This thesis is organised as follows. Chapter 2 provides an overview of wavelets and image compression. Chapter 3 gives the details of video compression and some of the latest works on video compression using the concept of motion vector. Chapter 4 describes a general algorithm which can be applied on any video and Chapter 5 explains the concept of macro motion vector and an algorithm for video compression restricted to video frames where camera is kept moving and the objects are fixed. Chapter 6 discusses the implementation issues, Chapter 7 shows the results and comparisons and Chapter 8 deals with conclusions, discussions and scope for further work.

Chapter 2

Wavelets for Image Compression

2.1 *What are wavelets*

Wavelets are mathematical tools that help remove the noise from the data. Wavelets were developed and are being used in the fields such as mathematics, quantum physics, electrical engineering, seismic geology, computer science. During the last few years, wavelets have found a large application area in the fields of data and image compression. The Wavelet Transform can be used to analyze the time and frequency content of an image and remove all redundancy as well as areas that cannot be perceived by the human eye. The human eye cannot resolve high frequencies, such as the human ear cannot hear high frequency sounds.

2.1.1 *Mother wavelet*

Wavelets constitute a family of functions derived from one single function and indexed by two labels one for position and one for frequency. This function is called the *mother wavelet*, denoted by ψ . Wavelets are defined as

$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-\tau}{s}\right) \quad (2.1)$$

where $s \neq 0$ is the scale factor, τ is the translation factor.

A Mother wavelet ψ should satisfy the following conditions:

$$\psi \in L^2(\mathbb{R}) \quad (2.2)$$

$$\int_{-\infty}^{\infty} \frac{|\Psi(\omega)|^2}{|\omega|} d\omega < +\infty \quad (2.3)$$

where Ψ is the Fourier transform of ψ defined by

$$\Psi(\xi) = \int_{-\infty}^{\infty} \psi(x)e^{-ix\xi} dx \quad (2.4)$$

Equation (2.2) implies that the mean value of the wavelet ψ should be zero i.e

$$\int_{-\infty}^{\infty} \psi(x)dx = 0 \quad (2.5)$$

and hence ψ must be oscillatory i.e ψ must be a wave. Thus a wavelet is a small wave which has its energy concentrated in time. It has the ability to allow simultaneous time and frequency analysis [3].

2.1.2 Wavelet transform

Any function $f(t)$ can be decomposed into a set of wavelets. There are two types of wavelet transform [4].

- **Continuous wavelet transform [5]**

An important method for analyzing the time and frequency contents of a function $f(t)$ by means of a wavelet is the continuous wavelet transform. The transform is done by multiplying the wavelet and $f(t)$, and calculating the integral of the product. The continuous wavelet transform of a function $f \in L^2(R)$ can be defined by

$$F(s, \tau) = \int f(t)\psi_{s,\tau}^*(t)dt \quad (2.6)$$

where ψ^* is the complex conjugate of ψ , $s \in R^*$ and $b \in R$.

The inverse wavelet transform is defined by

$$f(t) = \int \int F(s, \tau)\psi_{s,\tau}(t)d\tau ds \quad (2.7)$$

The continuous wavelet transform as described above cannot be used directly due to the following reasons [5].

1. The wavelet transform is calculated by continuously shifting a continuously scalable function over a signal and calculating the correlation between the two. The obtained wavelet coefficients are highly redundant as these scaled functions will be nowhere near an orthogonal basis. This redundancy is not agreeable for practical applications.
2. The number of wavelets used in the transform is infinite and this number should be reduced.

3. For most functions the wavelet transforms have no analytical solutions and they can be calculated only numerically or by an optical analog computer.

• **Discrete wavelet transform [5]**

To overcome the problems in the continuous wavelet transform, discrete wavelets have been introduced. Discrete wavelets are not continuously scalable and translatable but can only be scaled and translated in discrete steps.

Discrete wavelets are obtained by modifying equation (2.1) to

$$\psi_{j,k}(t) = s_0^{-j/2} \psi(s_0^j t - k\tau_0) \quad (2.8)$$

where s_0 is called the dilation factor and τ_0 is called the translation factor. The effect of discretizing the wavelet is that the time-scale space is now sampled at discrete intervals. We usually choose $s_0=2$ and $\tau_0=1$ so that the sampling of the frequency axis as well as time axis corresponds to dyadic sampling. Thus equation (2.8) becomes

$$\psi_{j,k}(t) = 2^{-j/2} \psi(2^j t - k) \quad (2.9)$$

The generation of the discrete wavelets and the calculation of the discrete wavelet transform is well matched to the digital computer.

The discrete wavelet transform is the discrete version of the continuous wavelet transform. A wavelet is represented by means of several filter coefficients and the transform is carried out by matrix multiplication. Any function $f(t)$ can be decomposed into a set of wavelets as

$$f(t) = \sum_{j,k} a_{j,k} \psi_{j,k}(t) \quad (2.10)$$

where the coefficients $a_{j,k}$ are calculated using the following formula.

$$a_{j,k} = \langle \psi_{j,k}(t), f(t) \rangle = \int f(t) \psi_{j,k}^*(t) dt \quad (2.11)$$

Equation (2.11) is the discrete wavelet transform of $f(t)$. When discrete wavelets are used to transform a continuous signal, the result will be a series of wavelet coefficients. Equation (2.10) is the inverse discrete wavelet transform which shows that any arbitrary signal can be reconstructed by summing the orthogonal wavelet basis functions, weighted by the wavelet transform coefficients. The mother wavelet ψ should be so chosen such that the wavelets $\{\psi_{j,k}(t)\}$ constitute an orthonormal basis of $L^2(R)$.

1.3 *Father wavelet*

Prior to the construction of ψ , one constructs a function ϕ such that the functions $\{\phi(t - k)\}$, $k \in Z$ constitute an orthonormal system. This orthonormal system can be supplemented to a full orthonormal basis of $L^2(R)$ with the functions $2^{j/2}\psi(2^j t - k)$, $j \in Z_+$, $k \in Z$ for some mother wavelet ψ .

This function ϕ is called the *father of the wavelets* or the *scaling function*.

2.1.4 *Multi resolution analysis*

4)

A multiresolution analysis of $L^2(R)$ is a sequence of closed subspaces $\dots, V_{-1}, V_0, V_1, V_2, \dots$ such that

1. $V_n \subset V_{n-1}$, $n \in Z$.
2. $\bigcup_{n=-\infty}^{\infty} V_n$ is dense in L^2 and $\bigcap_{n=-\infty}^{\infty} V_n = \{0\}$.
3. $f(x) \in V_n \Leftrightarrow f(2x) \in V_{n-1}$.
4. $f(x) \in V_0 \Leftrightarrow f(x - k) \in V_0$, for all $k \in Z$.
5. There exists a function $g \in V_0$ such that the collection $g(\cdot - k)$, $k \in Z$ is a Riesz basis for V_0 .

This means that if a set of signals can be represented by a weighted sum of $g(t - k)$ then a larger set including the original can be represented by a weighted sum of $g(2t - k)$. In other words, if the basic expansion signals are made half as wide and translated in steps half as wide, they will represent a larger class of signals exactly or give a better approximation of any signal.

2.1.5 *How to construct Wavelets*

In this section we will see how wavelets can be constructed from a scaling function ϕ [4]. We define a set of scaling functions in terms of integer translates of ϕ by

$$\phi_{0n}(x) = \phi(x - n) \tag{2.12}$$

for $n \in Z$ and $\phi \in L^2$. In wavelet theory it will be assumed that V_0 is generated by $\{\phi_{0n}\}$. Since $\phi \in V_0$ and $V_0 \subset V_{-1}$ we have $\phi \in V_{-1}$. But property (3) of multiresolution analysis implies that $\phi(2^{-1}\cdot) \in V_0$. Thus

$$\phi\left(\frac{1}{2}x\right) = \sum_{n=-\infty}^{\infty} c_n \phi(x - n), \quad x \in R \tag{2.13}$$

for some coefficients $\{c_n\}$. This is called the dilation equation. It can be rewritten as

$$\phi(x) = \sqrt{2} \sum_{n=-\infty}^{\infty} h_n \phi(2x - n) \quad (2.14)$$

The coefficients h_n are called the filter coefficients of the function ϕ which satisfy the following properties.

$$\sum_{n=-\infty}^{\infty} h_n = \sqrt{2} \quad (2.15)$$

$$\sum_{n=-\infty}^{\infty} (-1)^n h_n = 0 \quad (2.16)$$

The associated mother wavelet ψ is then generated through ϕ by the definition

$$\psi(x) = \sqrt{2} \sum_{n=-\infty}^{\infty} g_n \phi(2x - n), \quad g_n = (-1)^n \overline{h_{1-n}} \quad (2.17)$$

and the associated wavelets are

$$\psi_{m,n}(x) = 2^{-\frac{1}{2}m} \psi(2^{-m}x - n), \quad m, n \in Z \quad (2.18)$$

Wavelets are generally expressed by means of their filter coefficients h_n .

2.2 Examples of Wavelets

Generally wavelets are represented by their filter coefficients $\{h_n\}$.

1. Haar Wavelet:

$$h_0 = \frac{1}{\sqrt{2}}$$

$$h_1 = \frac{1}{\sqrt{2}}$$

2. Daubechies Wavelets

D4 :

$$h_0 = \frac{\sqrt{2}}{8} (1 + \sqrt{3})$$

$$h_1 = \frac{\sqrt{2}}{8} (3 + \sqrt{3})$$

$$h_2 = \frac{\sqrt{2}}{8} (3 - \sqrt{3})$$

$$h_3 = \frac{\sqrt{2}}{8} (1 - \sqrt{3})$$

D6 :

$$\begin{aligned}
 h_0 &= \frac{\sqrt{2}}{32}(b + c) \\
 h_1 &= \frac{\sqrt{2}}{32}(2a + 3b + 3c) \\
 h_2 &= \frac{\sqrt{2}}{32}(6a + 4b + 2c) \\
 h_3 &= \frac{\sqrt{2}}{32}(6a + 4b - 2c) \\
 h_4 &= \frac{\sqrt{2}}{32}(2a + 3b - 3c) \\
 h_5 &= \frac{\sqrt{2}}{32}(b - c)
 \end{aligned}$$

where $a = 1 - \sqrt{10}$, $b = 1 + \sqrt{10}$, $c = \sqrt{5 + 2\sqrt{10}}$

More Daubechies wavelets can be found in [6].

More information about wavelets can be found in [7].

2.3 Wavelets chosen for this project

The ideal wavelet would be compact, orthogonal, symmetric and continuous. The first symmetrical and orthogonal wavelet discovered was a simple square wave which is not continuous and it is known as the *Haar wavelet*. The scaling function and the wavelet function of Haar Wavelet are given below.

Scaling function $\phi(x) = \begin{cases} 1, & \text{if } 0 \leq x \leq 1 \\ 0, & \text{otherwise} \end{cases}$

Wavelet function $\psi(x) = \begin{cases} 1, & \text{if } 0 \leq x < \frac{1}{2}, \\ -1, & \text{if } \frac{1}{2} \leq x < 1, \\ 0, & \text{otherwise} \end{cases}$

Wavelet coefficients $h_0 = h_1 = \frac{1}{\sqrt{2}}$

2.4 Why wavelets for Compression

Digital data, i.e., data from speech, images, video and graphics, are highly correlated, and contain redundancy. Wavelet exploits this structure by providing a system by which this type of digital data can be represented accurately with a few parameters. Moreover, the computation involved in obtaining the representation is fast and efficient, usually linear in complexity. Because of this property, wavelet has been used in image compression, video compression, data compression, data transmission, geometric modeling, as well as in numerical computations.

2.5 Image Compression

An enormous amount of data is produced when a 2-D light intensity function is sampled and quantized to create a digital image. The amount of data generated may be so great that it results in impractical storage, processing, and communications requirements. Image compression addresses the problem of reducing the amount of data required to represent a digital image. Compression is the process used to reduce the physical size of a block of information. The three basic redundancies in digital image compression are coding redundancy, interpixel redundancy, psychovisual redundancy. Compression will be achieved when one or more of these redundancies are reduced or eliminated.

A compression system consists of two distinct structural blocks : an *encoder* and a *decoder*. An input image $f(x,y)$ is fed into the encoder which creates a set of symbols from the input data. After transmission over the channel, the encoded representation is fed to the decoder, where a reconstructed output image $\hat{f}(x,y)$ is generated. In general, $\hat{f}(x,y)$ may or may not be an exact replica of $f(x,y)$. Compression may be lossy or lossless. Image compression and video compression are generally lossy, but text compression, where the loss of even one character may result in ambiguous text, is lossless. There are basically three steps in encoding an image.

1. Image preprocessing : An efficient coding scheme can be designed by taking into account the spatial correlations which occur in natural images. In order to use the spatial correlations, the image must be transformed or decomposed in such a way that the correlation between the parameters describing the image is removed.
2. Quantization : It is used to further reduce the values of transformed coefficients in order to produce more zero coefficients.
3. Entropy coding : It reduces the number of bits required to send the image.

Details about these techniques can be found in any of the Image Compression text books [8] [9] .

2.6 Image Compression using wavelets

A good image compression scheme requires a good image decomposition scheme. We now come to the description of an efficient decomposition scheme based on multiresolution wavelet bases[4]. First the single dimensional case is described in this regard as an example. Image compression using wavelets is described by Mallat [10] and has been worked out by Daubechies [11]. We will follow the treatment and notation of Daubechies.

Let ϕ be the scaling function of a multiresolution wavelet bases.

$$\phi_{m,n}(x) = 2^{-m/2}\phi(2^{-m}x - n) \quad (2.19)$$

The spaces $V_m = \text{span}\{ \phi_{m,n} | n \in Z \}$ correspond to the different resolution levels of decomposition. Let ψ be a wavelet function and $\psi_{m,n}$ be the wavelets as defined in equation (2.9). Let W_m be the orthogonal complement of V_m in V_{m-1} , in other words,

$$V_{m-1} = V_m \oplus W_m$$

Let P_m and Q_m denote the orthogonal projections on V_m and W_m respectively. Let the sequence $(c_n)_{n \in Z}$ be the signal to be compressed. Define a sequence $(c_n^0)_{n \in Z}$ with $c_n^0 = c_n$. Let $f \in V_0$ be defined by

$$f(x) = \sum_n c_n^0 \phi_{0,n}(x) \tag{2.20}$$

A multiresolution analysis is applied to f in the following manner. The function f can be written as

$$f = P_1 f + Q_1 f$$

The first term is the low resolution representation of f , contained in V_1 , whereas the second term is the difference signal, contained in W_1 . Since the space V_1 is spanned by $\{ \phi_{1,n} | n \in Z \}$ we can write

$$P_1 f = \sum_k c_k^1 \phi_{1k}$$

where

$$c_k^1 = \langle P_1 f, \phi_{1k} \rangle = \langle f, \phi_{1k} \rangle = \sum_n c_n^0 \langle \phi_{0n}, \phi_{1k} \rangle$$

i.e

$$c_k^1 = \sum_n c_n^0 h_{n-2k}$$

where

$$h_n = 2^{-1/2} \int \phi(\frac{x}{2}) \phi(x - n) dx$$

Similarly we can write

$$Q_1 f = \sum_k d_k^1 \psi_{1k}$$

Then

$$d_k^1 = \sum_n c_n^0 g_{n-2k}$$

where

$$g_n = 2^{-1/2} \int \psi(\frac{x}{2}) \phi(x - n) dx$$

It is possible to choose ψ in such a way that

$$g_n = (-1)^n \bar{h}_{1-n}$$

As we choose ϕ to be real valued, this implies

$$g_n = (-1)^n h_{1-n}$$

By repeating this procedure for N times we arrive at the decomposition

$$f = Q_1 f + Q_2 f + \dots + Q_N f + P_N f$$

where

$$P_n f = \sum_k c_k^n \phi_{nk}$$

and

$$Q_n f = \sum_k d_k^n \psi_{nk}$$

The coefficients c_k^n and d_k^n are given by

$$c_k^n = \sum_m c_m^{n-1} h_{m-2k}$$

and

$$d_k^n = \sum_m c_m^{n-1} g_{m-2k}$$

These operations can be performed easily and are suitable for implementation in hardware. After a number of iterations, the original sequence c^0 is decomposed into a lowest resolution signal c^N and difference signals d^N, d^{N-1}, \dots, d^1 of finer and finer resolution.

The original sequence can be reconstructed by repeated use of the relation

$$c_n^{j-1} = \sum_k c_k^j h_{n-2k} + \sum_k d_k^j g_{n-2k}$$

Two dimensional case is a straightforward extension of one dimensional case. A wavelet transform is done on a square image in the following manner.

Suppose we have an $N \times N$ image. First each of the N rows is split into a low pass half and a high pass half. The wavelet transform on a row is explained in one dimensional case. The result is an $N \times \frac{N}{2}$ low pass sub-image and $N \times \frac{N}{2}$ high pass subimage. Next each column of subimages is split into a low pass half and a high pass half. The result is a four-way partition of the image into

1. LL : horizontal low-pass/vertical low-pass

LL portion gives the coarse scale information.

2. HL : horizontal high-pass/vertical low-pass
HL portion gives information about the vertical edges.
3. LH : horizontal low-pass/vertical high-pass
LH portion gives information about the horizontal edges.
4. HH : horizontal high-pass/vertical high-pass
HH portion gives information about the corner points.

This four-way partition of an image is shown below.

LL	LH
HL	HH

The multiresolution decomposition described above depends on the choice of an appropriate scaling function ϕ . Yet the algorithm uses only the coefficients h_n and g_n . Thus in section (2.2), wavelets are defined using the filter coefficients h_n . Daubechies has derived conditions that h_n and g_n must satisfy in order to make the algorithm work, without any reference to the function ϕ . These conditions are

1. $\sum_n |h_n| < \infty, \quad \sum_n |g_n| < \infty;$
2. $\sum_n \{h_{n-2k}h_{n-2l} + g_{n-2k}g_{n-2l}\} = \delta_{kl};$
3. $\sum_n h_{n-2k}g_{n-2l} = 0;$
4. $\sum_n h_n = \sqrt{2}, \quad \sum_n g_n = 0;$

The second and third conditions guarantee that the original image can be reconstructed from the decomposition and the decomposition contains no redundant information. The fourth condition indicates that H acts as an averaging operator, while G acts like a local detail filter.

2.7 Still Image Compression techniques using wavelets - JPEG 2000

JPEG has been designed as a compression method for continuous-tone images. The word JPEG is an acronym that stands for *Joint Photographic Experts Group*. The JPEG compression is based on Discrete cosine transform (DCT) and it has proved to be successful and has become widely used for image compression, especially in web pages. The use of DCT on 8×8 blocks of pixels results sometimes in a reconstructed image that has a blocky appearance. This is why the JPEG committee has decided, as early as 1995, to develop a new, wavelet based standard for the compression of still images, to be known as JPEG 2000 or JPEG Y2K. The following paragraph is a short summary of how the JPEG 2000 algorithm works [12].

Given an image it is partitioned into rectangular, nonoverlapping regions called tiles, that are compressed individually. If the image being compressed is in color, it is divided into three components. A tile is compressed in four main steps.

- Compute a wavelet transform that results in subbands of wavelet coefficients. Two such transforms, an integer and a floating-point, are specified by the standard. There are $L+1$ resolution levels of subbands, where L is a parameter determined by the encoder.
- The wavelet coefficients are quantized. This is done if the user specifies a target bit rate. The lower the bitrate, the coarser the wavelet coefficients have to be quantized.
- Use MQ coder to arithmetically encode the wavelet coefficients. The EBCOT algorithm has been adopted for the encoding step. The principle of EBCOT is to divide each subband into blocks, called code-blocks, that are coded individually. The bits resulting from coding several code-blocks become a packet and the packets are the components of the bitstream.
- The last step is to construct the bitstream. The markers can be used by the decoder to skip certain areas of the bitstream and to reach certain points quickly. Using markers, the decoder can for e.g., decode certain code-blocks before others, thereby displaying certain regions of the image before other regions. The bitstream is organized in layers, where each layer contains higher-resolution image information. Thus decoding the image layer by layer is a natural way to achieve progressive image transmission and decompression.

The details about these steps can be found in [2]

Current experiments indicates that JPEG 2000 performs better than the original JPEG, especially for images where very low bitrates or very high image quality are required. For lossless or near-lossless compression, JPEG 2000 offers only modest improvements over JPEG.

Chapter 3

Video Compression

3.1 *Video Compression - an Introduction*

The increasing demand to incorporate video data into telecommunication services, the corporate environment, the entertainment industry and even at home has made digital video technology a necessity. A problem, however, is that still image and digital video data rates are very large, typically in the range of 150 Mbits/sec. Data rates of this magnitude would consume a lot of the bandwidth, storage and computing resources in the typical personal computer. For this reason, video compression standards have been developed to eliminate picture redundancy, allowing video information to be transmitted and stored in a compact and efficient manner [15].

3.2 *Methods for Video Compression*

Video Compression is based on two principles. The first is the *spatial redundancy* that exists in each frame. The second is the fact that most of the time, a video frame is very similar to its immediate neighbors. This is called *temporal redundancy*.

Video compression is generally lossy. Encoding a frame F_i in terms of its predecessor F_{i-1} introduces some distortions. As a result, encoding frame F_{i+1} in terms of F_i increases the distortion. Even in lossless video compression, a frame may lose some bits. This may happen during transmission. If a frame F_i has lost some bits, then all the frames following it, up to the next intra frame, are decoded improperly, perhaps even leading to accumulated errors. This is why intra frames should be used from time to time inside a sequence, not just at its beginning. Any method for video compression follows this strategy [2].

In video compression the encoder may be slow, but the decoder has to be fast. A typical case is video recorded on a hard disk or on a DVD, to be played back. The encoder can take minutes or hours to encode the data. The decoder, however, has to play it back at the correct frame rate, so it

has to be fast. This is why a typical video decoder works in parallel. It has several decoding circuits working simultaneously on several frames.

3.3 *What is a motion vector*

In Video images there is not much of a difference between two consecutive frames as the objects or the camera or both are moving. If a part P of the preceding frame has been rigidly moved to a different location in the current frame then P can be compressed by writing the following three items on the compressed stream: its previous location, its current location and information identifying the boundaries of P.

In principle such a part can have any shape. In practice we are limited to equal-size blocks. The encoder scans the current frame block by block. For each block B it searches the preceding frame for an identical block C (if compression is to be lossless) or for a similar one (it can be lossy). Finding such a block, the encoder writes the difference between its past and present locations on the output. This difference is of the form

$$(C_x - B_x, C_y - B_y) = (\delta_x, \delta_y)$$

This is called the *motion vector* for that block.

3.4 *Video Compression Standards - using motion vector*

In this section we discuss the international video compression standards for video conferencing and multimedia applications [2].

3.4.1 *The H.261 Standard*

In late 1984, the CCITT¹ organized an expert group to develop a standard for visual telephony for ISDN services. The idea was to send images and sound between two terminals, so that users could talk and see each other. These are the videophone services and videoconferencing services. This type of services require sending large amounts of data, so compression became an important consideration. The group eventually came up with a number of standards, known as the H series, all operating at speeds of $p \times 64$ Kbit/s for $p = 1, 2, \dots, 30$. Video conferencing services generally require higher image quality, which can be achieved with $p \geq 6$.

The H.261 standard offers two important features:

¹CCITT : The International Telegraph and Telephone Consultative Committee

- It specifies a maximum coding delay of 150 msec. because it is mainly intended for bidirectional video communication. It has been determined that delays exceeding 150 msec. do not give the viewer the impression of direct visual feedback.
- It is amenable to low-cost VLSI implementation, which is rather important for widespread commercialization of videophone and teleconferencing equipment [1].

Video Multiplex - Data Structure used

The Video Multiplex defines a data structure so that a decoder can interpret the received bit stream without any ambiguity. The video data is arranged in a hierarchical structure consisting of a picture layer, which is divided into several group-of-block (GOB) layers. Each GOB layer in turn consists of macroblocks (MB), which are made of blocks of pixels. Each layer has a header identifying set of parameters used by the encoder in generating the bitstream that follows. The details about these layers can be found in [1].

Video Compression Algorithm [1]

The video compression scheme chosen for the H.261 standard has two main modes : the intra and inter modes. The intra mode is similar to JPEG still-image compression, which is based on a block-by-block DCT coding. In the inter mode, first a temporal prediction is employed with or without motion compensation. Then the interframe prediction error is DCT encoded. The algorithm can be summarized through the following steps.

1. Estimate a motion vector for each macroblock (MB). The standard does not specify a motion estimation method but block matching based on 16×16 luminance blocks is generally used. The decoder accepts one integer-valued motion vector per MB whose components do not exceed ± 15 .
2. Select a compression mode for each MB, based on a criterion involving the displaced block difference (dbd), which is defined by

$$dbd(x, k) = b(x, k) - b(x - d, k - 1) \quad (3.1)$$

where $b(.,.)$ denotes a block, x and k are pixel coordinates and time index, respectively, and d is the motion vector defined for the k th frame relative to the $(k-1)$ st frame. If d is set equal to zero, then dbd becomes a block difference (bd). We will discuss about these compression modes later.

3. Process each MB to generate a header followed by a data bitstream that is consistent with the compression mode chosen.

Selecting a compression mode requires making some of the decisions which are given below.

- Should a motion vector be transmitted.
- Inter vs. intra compression.
- Should the quantizer stepsize be changed.

Some of the possible compression modes are

- Intra : intraframe.
- Inter : interframe with zero motion vector.
- Inter+MC : motion-compensated interframe.

In order to select the best compression mode, at each MB, the variance of the original macroblock, the macroblock difference (bd), and the displaced macroblock difference (dbd) with the best motion vector estimate are compared as follows.

1. If the variance of dbd is smaller than bd as determined by a threshold, then the "Inter+MC" mode is selected, and the motion vector needs to be transmitted as side information. The difference of the motion vector between the present and previous macroblocks is variable length coded for transmission.
2. Otherwise, a motion vector will not be transmitted, and a decision needs to be made between the "Inter" and "Intra" modes. If the original MB has a smaller variance, then the "Intra" mode is selected, where the DCT of each 8×8 block of the original picture elements are computed, otherwise, the "Inter" mode (with zero displacement vector) is selected. In both "Inter" and "Inter+MC" blocks the respective difference blocks are DCT encoded.

A variable thresholding is applied before quantization to increase the number of zero coefficients. The coefficient values after variable thresholding are quantized using a uniform quantizer. Within a macroblock the same quantizer is used for all coefficients except for the intra DC coefficients. The intra DC coefficient is linearly quantized. In order to increase the coding efficiency, the quantized coefficients are zigzag scanned and events are defined as a combination of a run length of zero coefficients preceding a nonzero coefficient, and the value (level) of the nonzero coefficient i.e.,

$$\text{EVENT} = (\text{RUN} , \text{LEVEL})$$

For 8×8 blocks, we have $0 \leq \text{RUN} \leq 64$ and the dynamic range of the nonzero coefficients, LEVEL, is $[-128g, 127g]$, where g is the quantizer step size. These events are then variable length coded and then transmitted.

H.261 was targeted for teleconferencing applications where motion is naturally more limited. Motion vectors are restricted to a range of ± 15 pixels. Accuracy is reduced since H.261 motion vectors

are restricted to integer-pel accuracy. Later standards like H.263 [13], MPEG were developed. H.263 was designed for low bit rate communication. The coding algorithm of H.263 is similar to that used by H.261 with some improvements and changes to improve performance and error recovery.

In this work, for the purpose of comparison results of Miss America, one of the H.263 video sequences, are used.

3.4.2 *The MPEG Standard*

MPEG is an acronym for *Moving Picture Experts Group*. The term also refers to the family of digital video compression standards and file formats developed by the group. MPEG generally produces better-quality video than competing formats, such as Video for Windows, Indeo and QuickTime. MPEG files can be decoded by special hardware or by software [1] [2].

MPEG achieves high compression rate by storing only the changes from one frame to another, instead of each entire frame. The video information is then encoded using DCT. MPEG uses a type of lossy compression, since some data is removed. But the loss of data is generally imperceptible to the human eye. MPEG is well suited for high-quality and multimedia systems. There are three major MPEG standards: MPEG-1, MPEG-2 and MPEG-4.

Note that there is no reference to MPEG-3. This is because it was originally anticipated that this standard would refer to HDTV applications, but it was found that minor extensions to the MPEG-2 standard would suffice for this higher bit-rate, higher resolution application, so work on a separate MPEG-3 standard was abandoned.

The most common implementations of the MPEG-1 standard provide a video resolution of 352-by-240 at 30 frames per second (fps). This produces video quality slightly below the quality of conventional VCR videos. MPEG-2 offers resolutions of 720x480 and 1280x720 at 60 fps, with full CD-quality audio. This is sufficient for all the major TV standards, including NTSC, and even HDTV.

MPEG-2 is used by DVD-ROMs. MPEG-2 can compress a 2 hour video into a few gigabytes. While decompressing an MPEG-2 data stream requires only modest computing power, encoding video in MPEG-2 format requires significantly more processing power.

MPEG-4 is a graphics and video compression algorithm standard that is based on MPEG-1 and MPEG-2 and Apple QuickTime technology. Wavelet-based MPEG-4 files are smaller than JPEG or QuickTime files, so they are designed to transmit video and images over a narrower bandwidth and can mix video with text, graphics and 2-D and 3-D animation layers.

Now we discuss about one of these standards which is MPEG-1. A typical compression method used by MPEG-1 is described below.

1. First, motion estimation is performed on each macroblock. In addition to motion estimation from just the preceding frame, MPEG allows for prediction from frames in the past or future or a combination of a past and future frame (with restrictions). Since objects in the frame may not be moving steadily from frame to frame, each macroblock is allowed to have up to two motion vectors, one relative to a past frame and another relative to a future frame. Note that to allow for predictions from future frames, the extra frames must be buffered and the sequence should be coded out-of-order. Motion estimation is allowed over a greater range up to ± 1023 .
2. Next, a four-way decision must be made. MPEG allows the prediction formed from the arithmetic difference between the current macroblock and an offset macroblock from a past frame, future frame or an average between past and future frame, to be coded. Sometimes it has to code the current macroblock from scratch.
3. Key frames, also called Intra or I frames, which do not allow any predicted macroblocks are coded periodically to allow for random access into the video stream. Forward predicted frames (called P frames) allows macroblocks predicted from past P frames of I frames or macroblocks coded from scratch. I frames and P frames are used as past and future frames for Bi-directional predicted frames (called B frames). B frames allow for all four types of macroblocks.
4. An 8×8 DCT is applied to each block in either the arithmetic difference or the current macroblock. MPEG uses both matrices and a scale factor for quantization. Since the DC bin is the most important, it is quantized to a fixed 8 bit scale.
5. The final stage of compression is the zig-zag scanning, run-length encoding and entropy coding. Like H.261, MPEG specifies fixed Huffman coding tables for entropy coding.

To decompress an MPEG frame each operation is performed in reverse except for motion estimation. Since the motion vectors and the decision are embedded in the compressed bit-stream, the MPEG de-compressor just needs to apply the motion vector offsets to retrieve the prediction from the past and/or future frames, if necessary. This works fine for video but linear editing programs have trouble editing frame by frame because of the incomplete frames.

MPEG-4 [14] is a new standard that extends previous previous MPEG standards to include support for user interaction and flexible display of multiple video bitstreams. This is an object-based video coding, in which a video image is represented as a set of regions of interest, also called video objects, that are coded independently. At the decoder, users decode, compose and manipulate video objects from one or more bitstreams in a single display. In this standard they used discrete wavelet transform to improve the quality of the images.

In this section, just an introduction to some of the various components of MPEG video compression has been provided. Textbooks have been written about individual techniques such as the discrete cosine transform. The reader is encouraged to search out more in-depth information on these topics, MPEG syntax, applications, etc.,

Intel has recently developed Indeo video interactive 4.0 (IVI). This is a wavelet based lossy compression scheme. It has a built-in scalability due to which both fast and slow processors will see complete movies without dropped frames. Faster processors will see better quality video.

3.5 *Why a new method*

In spite of having all these standards, a new method is needed to be developed because of the following reasons.

- Since this field has enormous applications, almost all of the methods developed were patented due to which only the outlines of the methods are described. It is not quite clear how the stated PSNR values and stated compression ratio values are obtained for several methods.
- As mentioned before there are 3 types of video depending on the movement of camera or scene or both. If one closely observes those images where the camera is moving and objects are fixed one can see that the difference between two consecutive frames is only at the corners as shown in the figure.



where the empty box represents the previous frame and the other one is the current frame.

These are the four cases one can observe while dealing with type 2 video images. Let us call these cases as RD, LD, RU, LU where L stands for left, R for right, D for down and U for up.

One can make use of this concept to attain a good compression ratio by using an efficient tool like wavelet and probably it may be extended to the case of type 3 video where both the objects and the camera are moving. This idea is the basic step for developing a new method. We will explore on this method in Chapter 5.

Chapter 4

Proposed wavelet based Video Compression method for any video

4.1 *Algorithm*

A typical technique for video compression should start by encoding the first frame using a still compression method. It should then encode each successive frame by identifying the differences between the frame and its predecessor, and encoding these differences. If the frame is very different from its predecessor it should be coded independently of any other frame. In the video compression literature, a frame that is coded using its predecessor is called inter frame, while a frame that is coded independently is called intra frame [17]. In this algorithm, which is proposed for any video, intra frame coding and inter frame coding are performed in the following way.

4.1.1 *Intra frame coding*

The steps that are performed in coding an intra frame are

1. Discrete wavelet transform (DWT)
2. Removal of HH portion

Let us discuss about these steps in detail.

Discrete wavelet transform

Much research effort has been expended in the area of wavelet-based compression with the results indicating that wavelet-based approaches outperform DCT-based techniques. However, it is not completely clear which wavelets are suitable for video compression. Wavelets implemented using

Linear-phase filters are generally advantageous for image processing because such filters preserve the location of spatial details.

The Wavelets that are used in this algorithm are *Haar Wavelets* whose filter coefficients are

$$h_0 = \frac{1}{\sqrt{2}}$$

$$h_1 = \frac{1}{\sqrt{2}}$$

and corresponding g_n 's are

$$g_0 = \frac{1}{\sqrt{2}}$$

$$g_1 = -\frac{1}{\sqrt{2}}$$

Using these wavelets a discrete wavelet transform (DWT) is applied on the intra frame. Details about DWT can be found in Section (2.1.3) which divides the image frame into 4 portions : LL, HL, LH and HH as discussed in Section (2.6).

Removal of HH portion

The HH portion of the transformed image contains information about the corner points and the diagonal edges. Majority of the information is present in the other 3 portions. So, generally the loss incurred by removing HH portion is negligible and not discernable to the naked eye. Thus HH portion of the image is removed and then transmitted to the other end.

Decoding the intraframe is very easy as the receiver will retrieve the replica of the image by performing the inverse wavelet transform on the image received.

4.1.2 Inter frame coding

The steps that are performed in coding an inter frame are given below.

1. Finding the difference image
2. DWT
3. Quantization
4. Modified Huffman coding
5. Modified run length coding

These steps are discussed below.

Finding the difference image

In video images it is the fact that very often, a video frame is very similar to its immediate neighbours. This is called *temporal redundancy* and it can be used to achieve a good compression. The prediction error, also called differential signal, is obtained by finding the difference between the current frame and its predecessor. This image is called the difference image.

DWT

Discrete wavelet transform is applied on this difference image which results in LL, HL, LH and HH portions.

Quantization [1]

Quantization is the process of representing a set of continuous-valued samples with a finite number of states. If each sample is quantized independently, then the process is known as scalar quantization. A scalar quantizer $Q(\cdot)$ is a function that is defined in terms of a finite set of decision levels d_i and reconstruction levels r_i , given by

$$Q(s) = r_i, \text{ if } s \in (d_{i-1}, d_i], \quad i = 1, \dots, L \quad (4.1)$$

where L is the number of output states. If all the reconstruction levels are equally spaced then it is called *uniform quantization*.

Uniform scalar quantization is used in this algorithm whose implementation details are given in section (6.2)

Modified Huffman coding [16]

Modified Huffman Coding (MHC) is a standard variable length coding technique where the input data is encoded by a data structure having two fields (range, pointer). Range x denotes that the input data is an element of the set S containing 2^x elements and pointer p indicates a specific value inside the set S . x is coded using Huffman coding and p is expressed as a binary number of x bits. This data structure is similar to the one used in JPEG. The values included in Range = 0,1,2,...,n are given below.

range	values to be considered
0	0
1	-1,1
2	-3,-2,2,3
3	-7,-6,-5,-4,4,5,6,7
.	.
n	$-2^n + 1, -2^n + 2, \dots, -2^{n-1}, 2^{n-1}, \dots, 2^n - 2, 2^n - 1$

Modified Run length coding

Run length coding (RLE) is done after entropy coding to reduce the number of bits used to represent the coded data. It is generally done as follows.

Data is encoded as (run, value) pairs, where run is the number of zero's or one's and value is the next non-zero component. This is useful if the data contains a long run of 0's or 1's. To achieve a good compression ratio, this technique has been modified as follows. Every run of length l is represented in the binary form, which reduces the total number of bits required to be sent. For example, the number of bits required to send a positive integer $n < 256$, if we represent the same n in binary form the number of bits required will be $\log n$. In this way RLE is applied on the Huffman coded data and sent to the other end.

Decoding an interframe is exactly the reverse of the above process.

4.2 Data sets

The datasets used for this algorithm are given below.

1. Traffic sequence : In this video camera is kept fixed and the objects are moving.
2. Hand sequence : This is the video of a hand holding a bowl in which camera is moving around the hand.
3. Ball sequence : This is the video of a tennis ball.

The implementation details are given in Chapter 6, and the results with the figures are given in Chapter 7.

Chapter 5

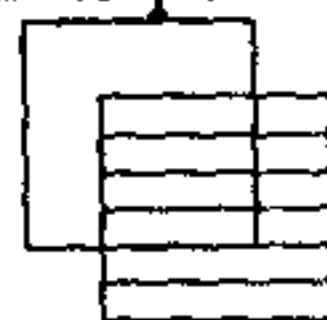
Proposed method for Video Compression using Macro motion vector.

The basic premise of motion estimation is that in most cases, consecutive video frames will be similar except for changes induced by objects moving within the frames. In the trivial case of zero motion between frames (and no other differences caused by noise, etc.), it is easy for the encoder to efficiently predict the current frame as a duplicate of the prediction frame. When this is done, the only information necessary to transmit to the decoder becomes the syntactic overhead necessary to reconstruct the picture from the original reference frame. When there is motion in the images, the situation is not as simple. We discussed about the concept of motion vector in Section (3.3) which is meant for all video images.

This chapter deals with type 2 video. This method is developed by using the fact that when camera is kept moving and objects are fixed, the difference between two consecutive frames is only at the corners as shown in the figure in section (3.5). Though this method is restricted to a particular type of data, it provides a good compression ratio with good quality. In this method, high quality video is maintained at a slight cost to coding efficiency.

5.1 *What is a Macro motion vector*

The concept of Macro motion vector is explained with an example. Consider the RD¹ case



¹This is discussed in section (3.5)

In this figure, the frame with horizontal lines is the current frame and the empty frame is the previous frame. The intersection part indicates those blocks in the current frame that match with the previous frame. Thus, to send the information about this common portion, the intersection region needs to be found. The information about this intersecting region can be found from the coordinates of the two corners, as the intersecting region needs to be a rectangle.

If one can find the coordinates of that point in the previous image from which the previous image is same as the current image then it is not necessary to send the intersection part. That is, instead of matching two consecutive frames block by block we are trying to find a Macro block² in the previous image that matches with the current image. The difference between the co-ordinates of these two macroblocks gives a motion vector for this Macro block. This is called the *Macro motion vector* (MMV).

Let B be the Macro block of the previous image which matches with the Macro block C of the current image. Let (B_x, B_y) and (C_x, C_y) be the coordinates of the left corner of Macro blocks B and C respectively. Then the Macro motion vector is defined by

$$MMV = (B_x - C_x, B_y - C_y) \quad (5.1)$$

So instead of sending the motion vector for every block, as done in many of the motion estimation methods, it is sufficient to send one MMV and its position in the current image for the matched portion. The same technique is applied to LD, RU and LU cases.

5.2 *Algorithm*

In this method intra frame and inter frame are coded in a different manner. So we will discuss about their coding schemes separately.

5.2.1 *Intra frame coding*

The steps involved in coding an intra frame are given below.

1. DWT
2. Quantization
3. Modified Huffman coding
4. Modified RLE

All these methods were discussed in detail in subsection (4.1.2).

²Macro block means the intersected block

5.2.2 *Inter frame coding*

The steps involved in coding an inter frame are :

1. Partitioning the image into blocks of size 8×8 .
2. Finding the Macro motion vector.
3. Coding the blocks.

The second and third steps are given below in detail.

Finding the Macro motion vector

In order to find a Macro motion vector we have to find a best block (BB). BB is an 8×8 block whose motion vector is suitable for maximum number of blocks.

Algorithm for finding a best block is given below.

1. while there are blocks in the image do
2. take a block C.
3. find a motion vector of that block.
4. if motion vector doesn't exist for C goto step 1.
5. else check whether majority of the blocks above and below C have the same motion vector.
6. if check fails goto step 1.
7. else check whether majority of the blocks on the left side and right side of C have the same motion vector.
8. if check fails goto step 1.
9. else check whether majority of the blocks which are diagonally opposite to C have the same motion vector.
10. if check fails goto step 1.
11. else return this as the best block.
12. if there are no more blocks return that there is no best block.

Once a best block is found its motion vector is treated as the Macro motion vector. If no such best block is found the conditions on the majority of the blocks to match will be relaxed and the algorithm is repeated.

Coding the blocks

Algorithm for coding a block B is given below.

1. Check whether the motion vector of the block B is MMV.
2. if check fails, find the motion vector of B.
3. if no motion vector exists then send the block as it is.
4. else send the motion vector.

This algorithm is repeated for all the blocks and accordingly either no information or its motion vector or the block as it is, will be sent. If a block has its motion vector as MMV it is not necessary to send the information as the first information that will be sent to the receiver is MMV.

5.3 Data sets

The datasets used to test this algorithm are given below

1. LFA sequence : This video is a sequence of buildings taken from a low flying aircraft (LFA).
2. Hand sequence : Video of a hand holding a bowl.
3. Book sequence : Video of a stack of books.
4. Lab sequence : Video of MIU lab, ISI.
5. Golf ball sequence : Video of a golf ball.

All these video are of type 2. The implementation details are discussed in Chapter 6, and the results with the figures are given in Chapter 7.

Chapter 6

Implementation details

6.1 Discrete Wavelet Transform(DWT)

DWT is implemented by using the matrix multiplication. Two types of filters are used to perform DWT. They are called the Haar_filter and the inverse_Haar_filter. Consider a square image I of order n . Then Haar_filter is also a square matrix of order n and it is defined as

$$\text{Haar_filter} = \begin{bmatrix} h_0 & h_1 & 0 & 0 & \dots & \dots & \dots & 0 \\ 0 & h_0 & h_1 & 0 & \dots & \dots & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \dots & \dots & h_0 & h_1 \\ h_1 & 0 & 0 & 0 & \dots & \dots & \dots & h_0 \\ g_0 & g_1 & 0 & 0 & \dots & \dots & \dots & 0 \\ 0 & g_0 & g_1 & 0 & \dots & \dots & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \dots & \dots & g_0 & g_1 \\ g_1 & 0 & 0 & 0 & \dots & \dots & \dots & g_0 \end{bmatrix}$$

Since h_n and g_n satisfy the conditions given in section (2.6), *inverse_Haar_filter* is just the transpose of the *Haar_filter* H .

Thus, discrete wavelet transform of I is implemented as

$$DWT(I) = HAH' \tag{6.1}$$

and the inverse wavelet transform of I is

$$IWT(I) = H'AH \quad (6.2)$$

6.2 Quantization

Quantization is done on every 8×8 block. Let X be an array containing all these 64 elements. Quantization is implemented using the following steps.

1. find the minimum value, min, of the block X.
2. for $i = 1$ to 64 do
3. $X[i] = X[i] - \text{min}$.
4. $X[i] = X[i] \gg 4^1$.
5. for $i = 2$ to 64 do
6. $X[i] = X[i] - X[i-1]$.

6.3 Modified Huffman and run length coding

In the original image, the number of bits required to represent a gray level is 8. Observe that after quantization the coefficients are in the range $[-31,31]$. The modified huffman code book² used in this algorithm is given below.

Range	Values	code for range
0	0	0
1	-1,1	10
2	-3,-2,2,3	110
3	-7,-6,-5,-4,4,5,6,7	1110
4	-15,-14,-13,-12,-11,-10,-9,-8,8,9,10,11,12,13,14,15	11110
5	-31,-30,-29,-28,-27,-26,-25,-24,-23,-22,-21,-20,-18,-17,-16,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31	11111

Using this table, any coefficient x is coded as follows.

¹ $N \gg 4$ is nothing but right shift N by 4 which is same as dividing N by 4

²code book is a look up table which gives a code for the corresponding coefficient

- Find the range r of x .
- Find the level l of x .
- code of x = binary representation of l followed by the code of r .

For example, suppose that the encoder finds -10 . Observe that the range r of -10 is 4 and -10 is the 6th element in that row. so level l is 6. Number of bits required to code level of an element in range r is r . Code of range 4 is 11110 and binary representation of 6 using 4 bits is 0110. Hence code of -10 is 011011110. This type of coding is dependent on the fact that the probability of getting a coefficient in the range 0,1,2 is very high. In this way we will be reducing the total number of bits for the whole image.

A modified run length coding is performed on the Huffman coded data. It is implemented using two counts `one_count` and `zero_count`. `One_count` is used to count the run of ones and `zero_count` is used to count the run of zeroes. This counting will stop if a number, different from the number in run, is found. This run count is sent in binary form, using the relevant number of bits required. It is done using the following table.

Range of run count	number of bits
1	1
2-5	2
6-13	3
14-29	4
30-61	5
else	6

For example, suppose that the run count is 32. Observe that, it is the third element in the range 30-61. Hence it is run length coded as third element using 5 bits i.e 00010. Similarly 33 is coded as 00011 and so on, 61 is coded as 11111. In this way a good compression ratio can be obtained.

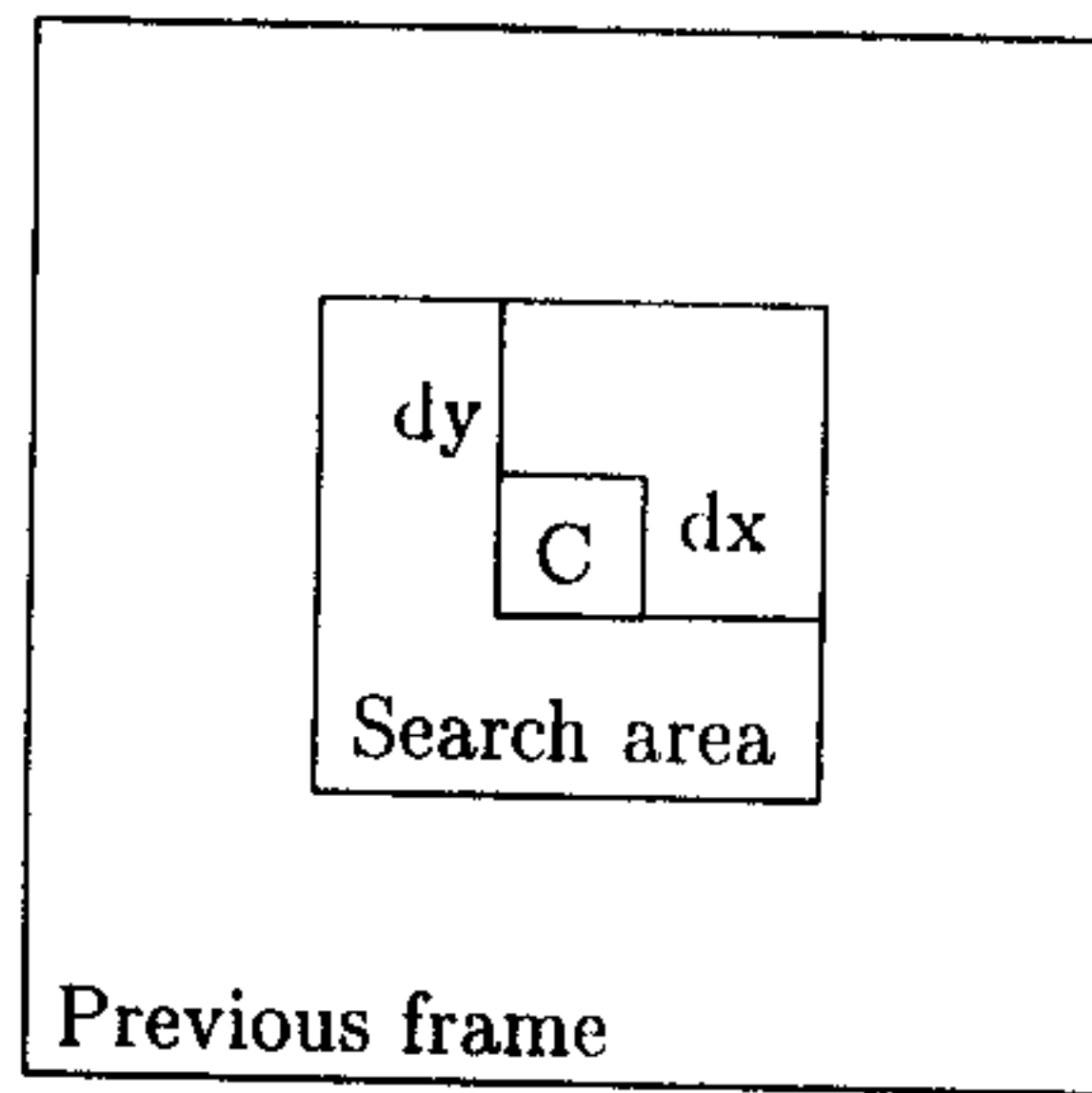
6.4 Finding a motion vector

In this algorithm *block search method* is used to find a motion vector. To find a motion vector for any block C in the current frame we have to find a block B in the previous frame which exactly matches with C . This is done using the *distortion measure*. This is the most sensitive part of the encoder. The distortion measure selects the best match for block C . It has to be fast and simple but also reliable. The distortion measure used in this algorithm is mean absolute difference (MAD) [2]. It

calculates the average of the absolute differences between a pixel $C(i,j)$ in C and its counterpart $B(i,j)$ in a candidate block B . It is defined as

$$MAD = \sum_{i,j} | C(i,j) - B(i,j) | \quad (6.3)$$

A block B in the previous frame, whose MAD is less, is chosen as the best block provided this MAD is less than some chosen threshold. It is not necessary to search the whole previous frame for a best match if we can take advantage of the temporal redundancy. Since there is not much difference between two consecutive frames, our search is restricted to some area around its counterpart in the previous frame. This is called the *search area*. The search area is defined by the maximum displacement parameters dx and dy . These parameters specify the maximum horizontal and vertical distances in pixels between C and the matching block in the previous frame. If C is a square with side b , the search area will contain $(b+2dx)(b+2dy)$ pixels and will consist of $(2dx+1)(2dy+1)$ distinct overlapping $b \times b$ squares. The number of candidate blocks in this area is therefore proportional to dx and dy . In this algorithm, dx and dy are both equal to 16.



Finding a motion vector of a block C is implemented as follows.

1. for $a = -16$ to 16
2. for $b = -16$ to 16 do
3. $B_x = C_x + a.$
4. $B_y = C_y + b.$
5. Find a block B in the previous image whose bottom left coordinates are $(B_x, B_y).$
6. Find MAD of $B.$

7. if minimum_MAD is greater than MAD then $M_x = B_x - C_x$, $M_y = B_y - C_y$.
8. else continue.
9. if minimum_MAD \leq THRESHOLD return (M_x, M_y)
10. else return that motion vector does not exist.

where minimum_MAD is the minimum value of the mean absolute difference of all the blocks in the search area and (M_x, M_y) are the coordinates of the motion vector.

Initial conditions :

1. minimum_MAD = ∞
2. $(M_x, M_y) = (0,0)$

6.5 *Finding a Macro motion vector (MMV)*

Prior to finding an MMV we should find the best block whose motion vector matches with maximum number of blocks. The implementation details of finding this best block is described below.

6.5.1 *Finding a best block*

1. Find the motion vector (mv_x, mv_y) of the current block C.
2. Increment C_x by 8 to obtain a block D, until all the blocks on the right side of C are exhausted, and find the motion vector of D. Increment block_count if D has the motion vector (mv_x, mv_y) else goto step 10.
3. Decrement C_x by 8 to obtain a block D, until all the blocks on the left side of C are exhausted, and find the motion vector of D. Increment block_count if D has the motion vector (mv_x, mv_y) else goto step 10.
4. If block_count is less than THRESHOLD_MATCHING_BLOCKS then goto step 10.
5. Increment C_y by 8 to obtain a block D, until all the blocks on the right side of C are exhausted, and find the motion vector of D. Increment block_count if D has the motion vector (mv_x, mv_y) else goto step 10.
6. Decrement C_y by 8 to obtain a block D, until all the blocks on the left side of C are exhausted, and find the motion vector of D. Increment block_count if D has the motion vector (mv_x, mv_y) else goto step 10.

7. If `block_count` is less than `THRESHOLD_MATCHING_BLOCKS` then goto step 10.
8. Increment or decrement C_x and C_y to obtain a block D, until all the blocks which are diagonally opposite to C are exhausted, and find the motion vector of D. Increment `block_count` if this block has the motion vector (mv_x, mv_y) else goto step 10.
9. If `block_count` \geq `THRESHOLD_MATCHING_BLOCKS` then return C as the best block and (mv_x, mv_y) as the MMV.
10. Increment or decrement C_x or C_y or both to get the next block of C in the current frame and goto step1.

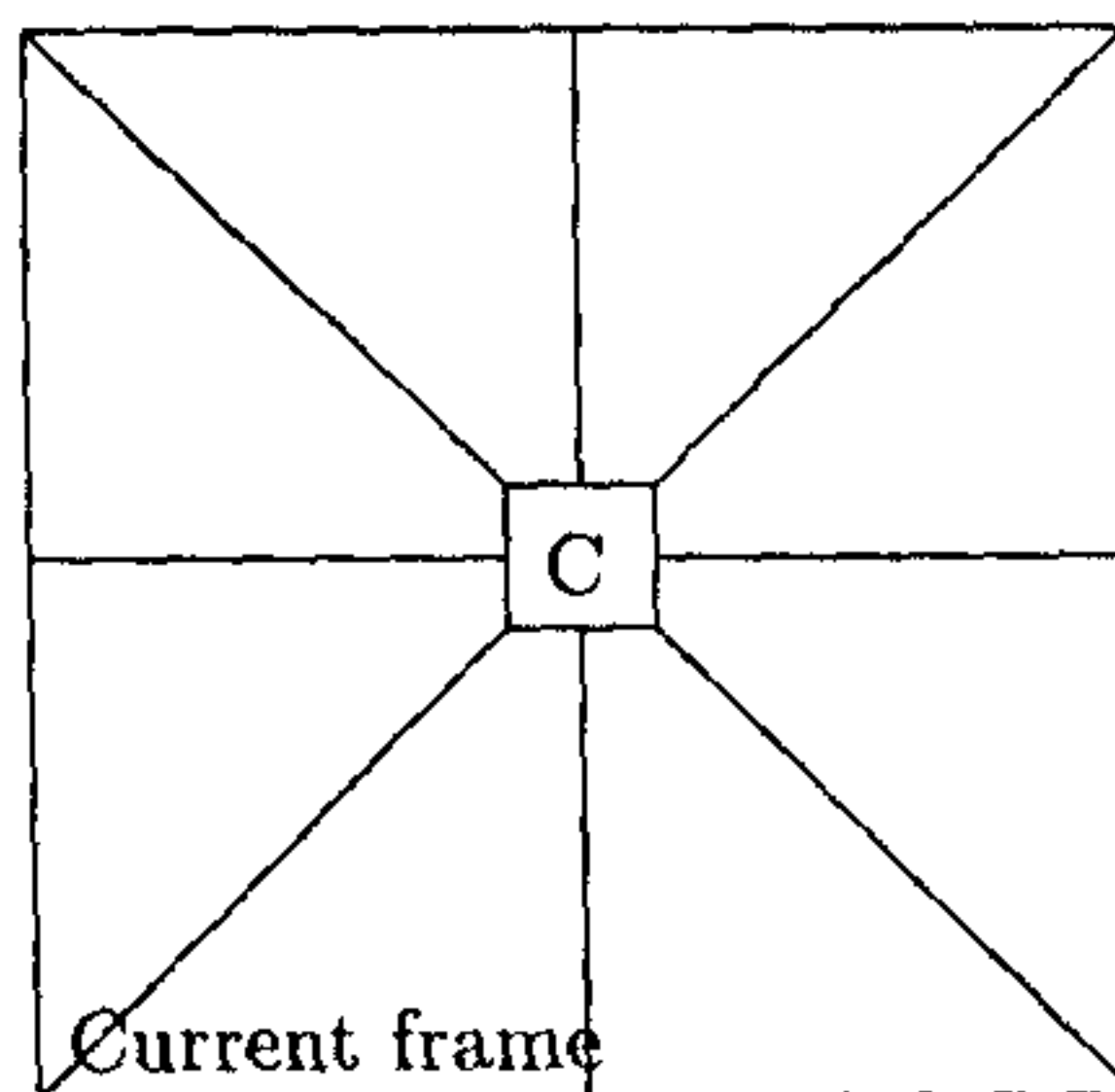
where

- (C_x, C_y) are the coordinates of the top left corner of a block C in the current frame,
- (mv_x, mv_y) are the coordinates of the motion vector and
- `block_count` = number of blocks whose motion vector is same as the motion vector of C.

Initial Conditions :

1. $C_x = 0$ and $C_y = 0$.
2. `THRESHOLD_MATCHING_BLOCKS` = 30.

Search for a best block in the current frame is done as shown in the following figure.



where lines in the figure indicate the blocks that are searched from the block C for the matching motion vector (mv_x, mv_y) .

6.6 Video Compression - Implementation

The first method of video compression is implemented in the following manner.

For every 5 consecutive video frames

1. Frame 1 is coded independently.
2. Frames 2 to 5 are coded in the following manner.
 - Frame 3 is coded using frame 1, and 2 is taken as the average of received frames 1 and 3.
 - Frame 5 is coded using frame 3, and 4 is taken as the average of received frames 3 and 5.

The second method of video compression is implemented in the following manner.

For every 10 consecutive video frames

1. Frame 1 is coded independently.
2. Frames 2 to 10 are coded using previous frames.

The compression ratio and the PSNR are found in the following manner.

Compression ratio is defined as the ratio of the size of the video that one starts with to the size of the video after it has been compressed. If every image is of order N (i.e. size of the image is $N \times N \times 8$) then compression ratio

$$CR = \frac{N \times N \times 8 \times \text{number of video frames sent}}{\text{total number of bits sent after compression}} \quad (6.4)$$

PSNR is an acronym for *Peak Signal to Noise Ratio*. For a single frame, PSNR is defined as

$$PSNR = 20 \log_{10} \left(\frac{255}{RMSE} \right) \quad (6.5)$$

where

$$RMSE = \frac{\sum_{i,j} [f(i,j) - F(i,j)]^2}{N^2} \quad (6.6)$$

is the root mean square error between the original frame f and the frame F obtained after compression.

Chapter 7

Results and Comparisons

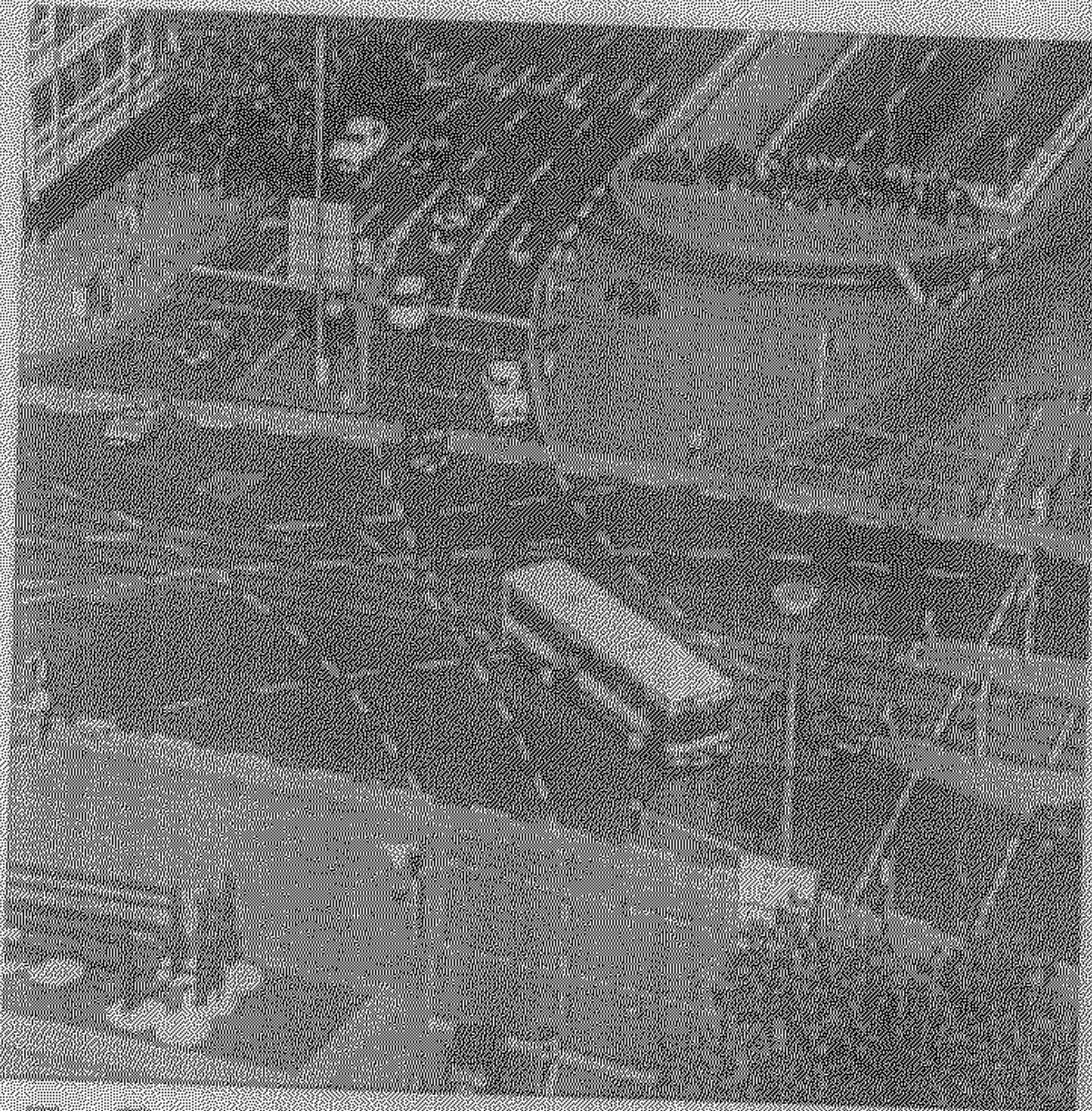
7.0.1 Results of First method

These are the results of the first method with the datasets mentioned in Section(4.2).

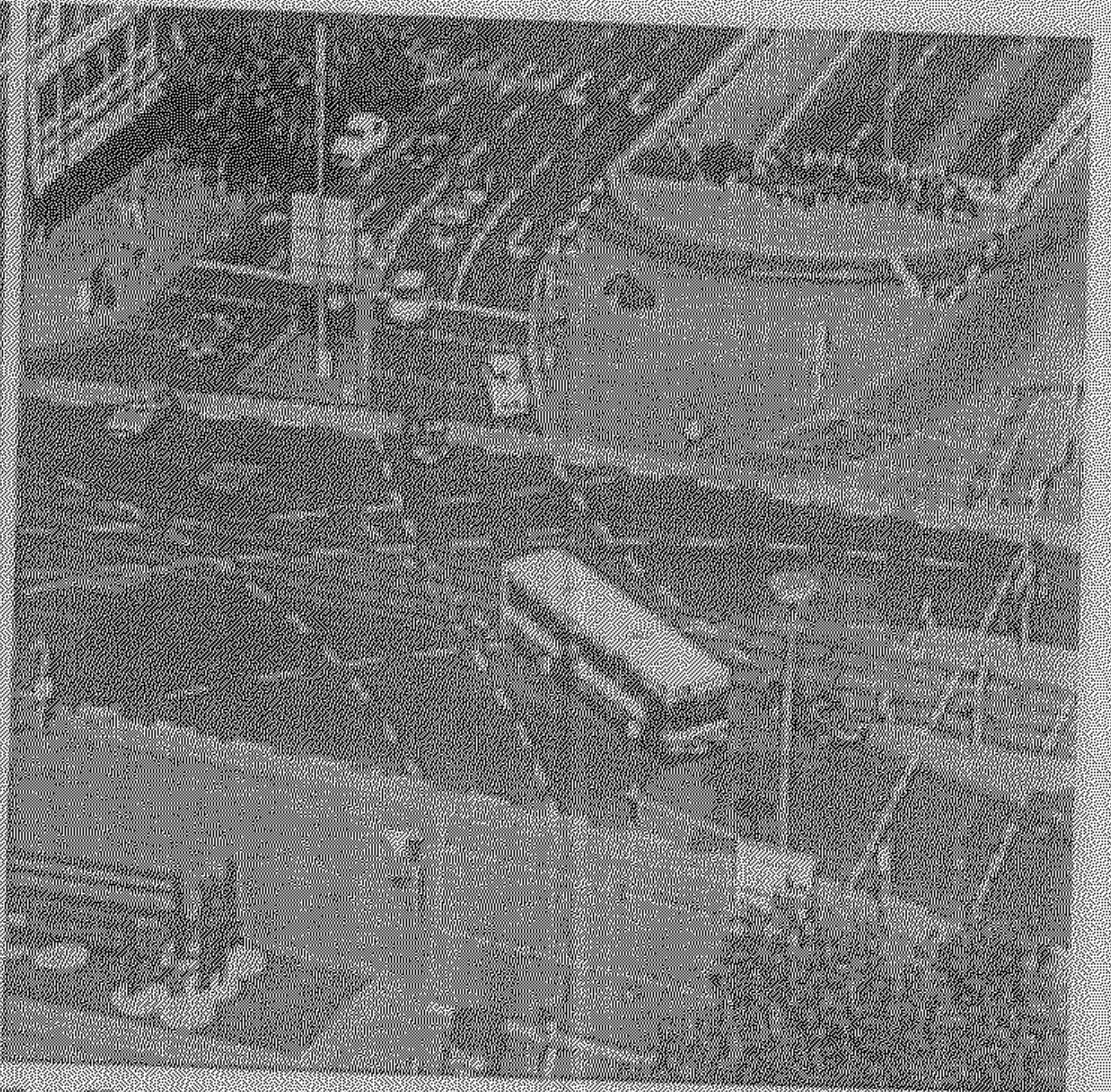
Dataset	PSNR	Compression ratio
Traffic	27.94	5:1
Hand	31.91	5:1
Tennis ball	33.43	6:1

From this table one can infer that though a good compression cannot be obtained, quality of the image is maintained. This technique is more useful in those areas where one wants to have a good quality.

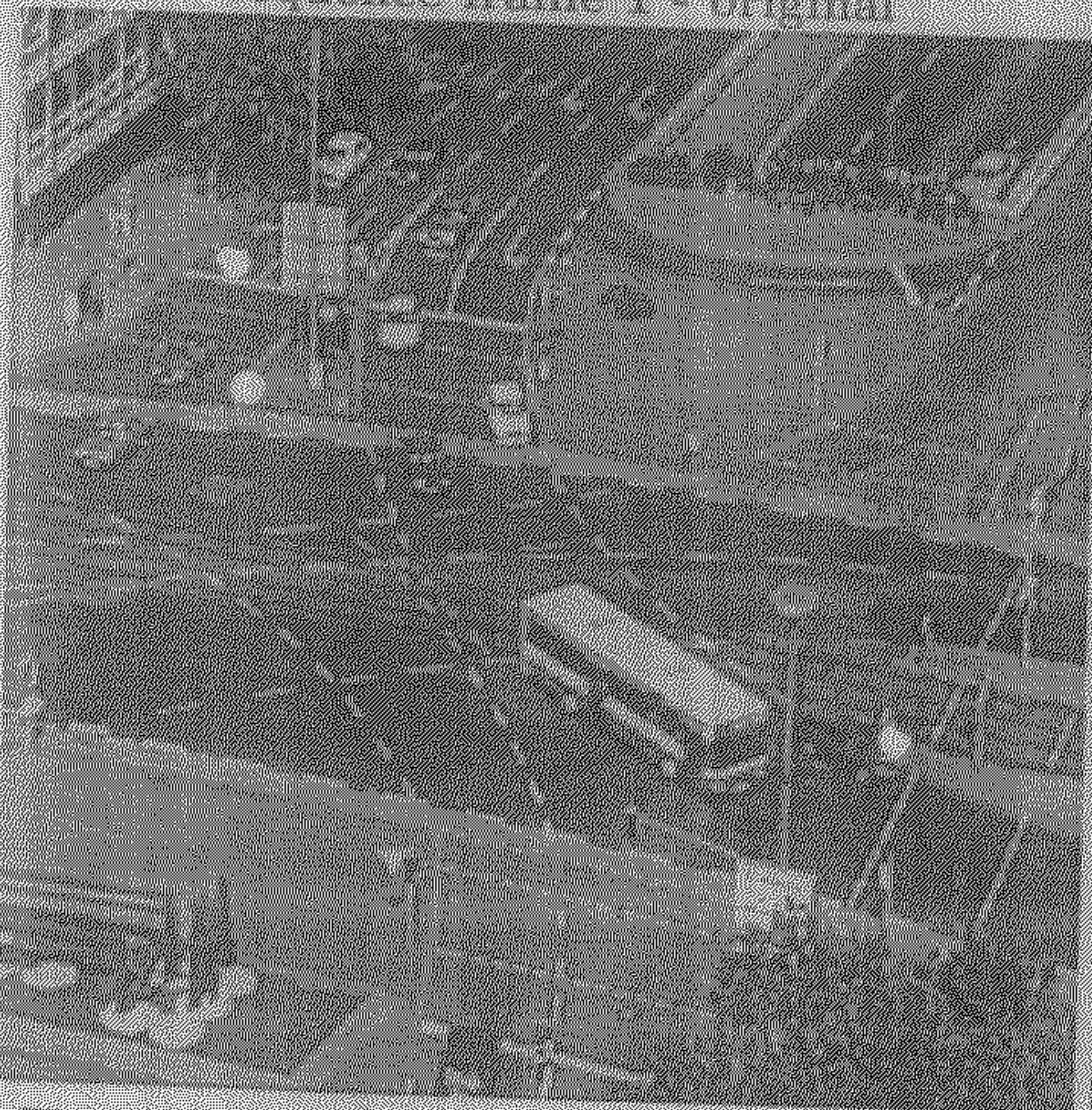
The images obtained after applying the first method on the data sets given in the above table are shown below. Since it is not possible to display the figures of all the images for every dataset, the images of the first frame and the fifth frame before compression and after compression are displayed. In this method, for every 5 frames of a dataset, the first frame is coded independently and the frames 2 to 5 are coded using the information about the previous frame.



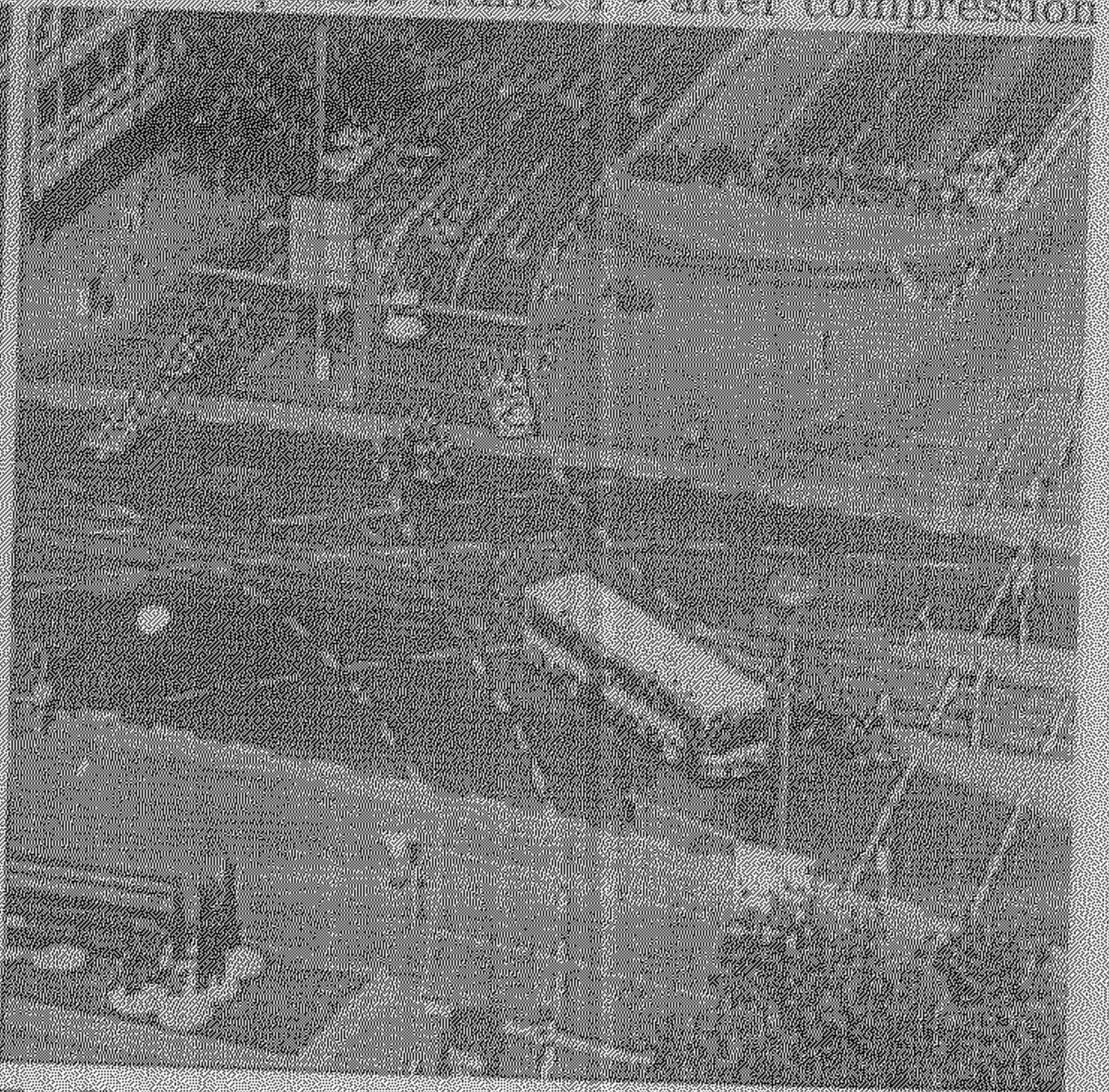
Traffic sequence frame 1 - original



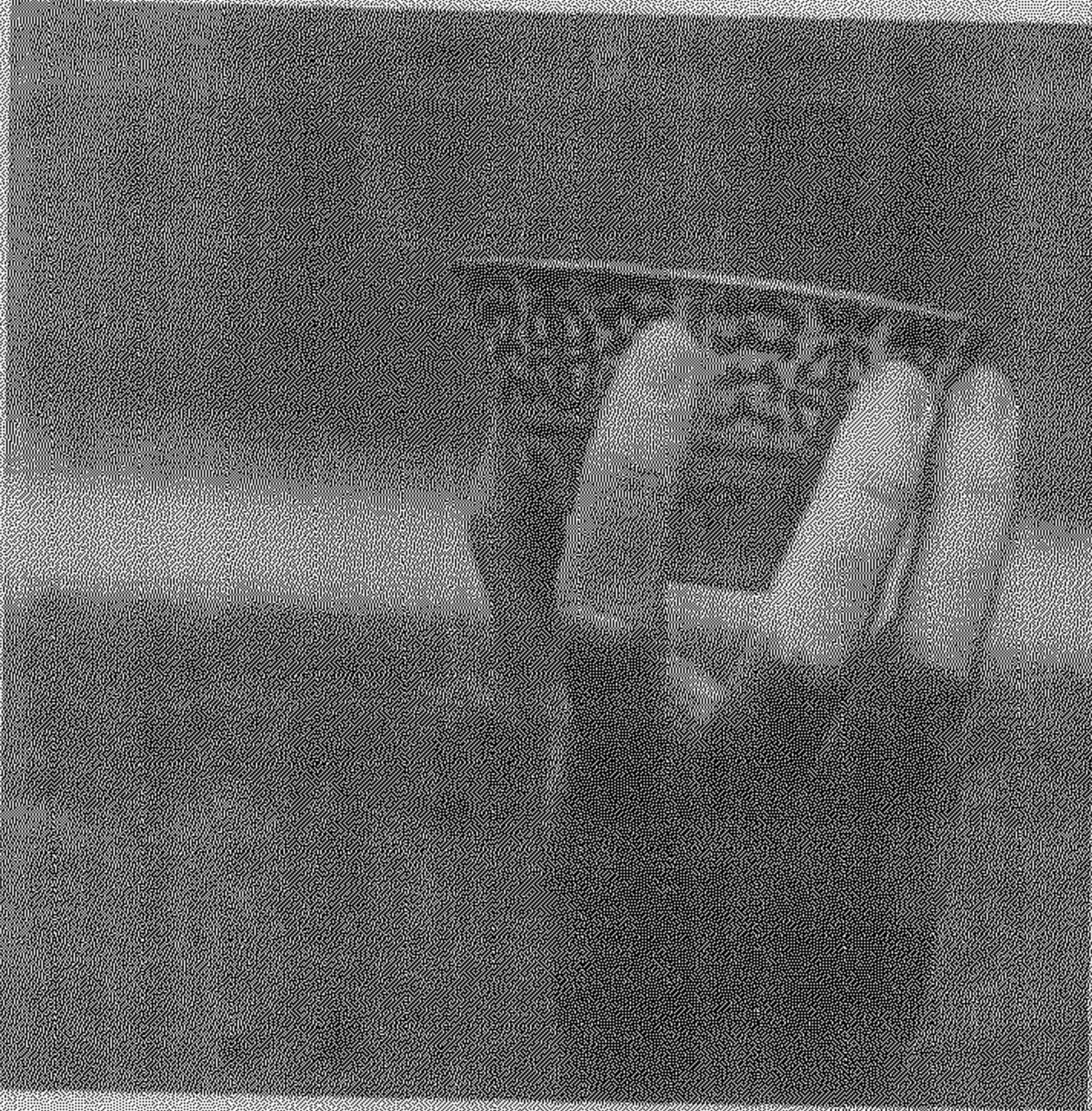
Traffic sequence frame 1 - after compression



Traffic sequence frame 5 - original



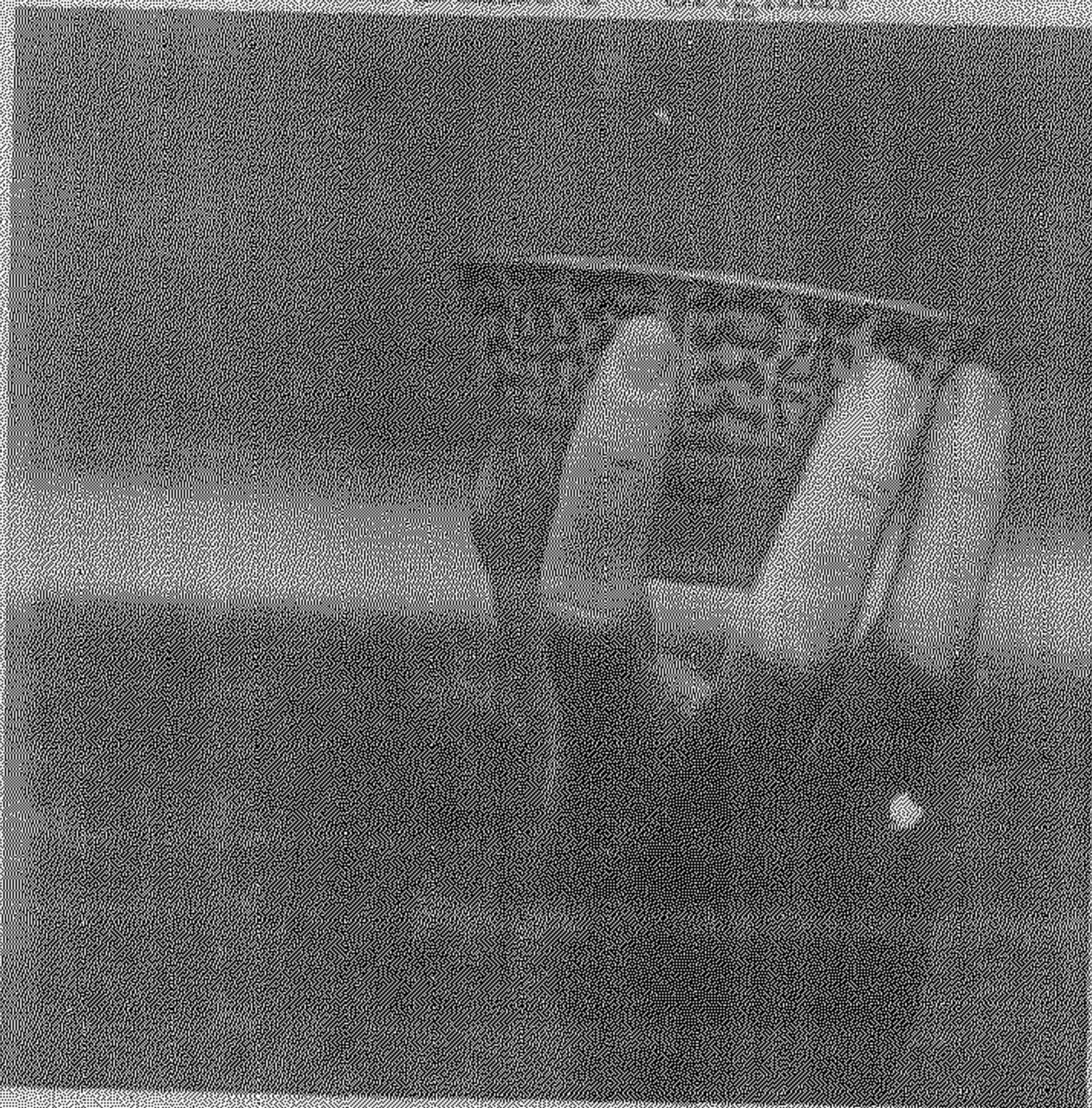
Traffic sequence frame 5 - after compression



Hand frame 1 - original



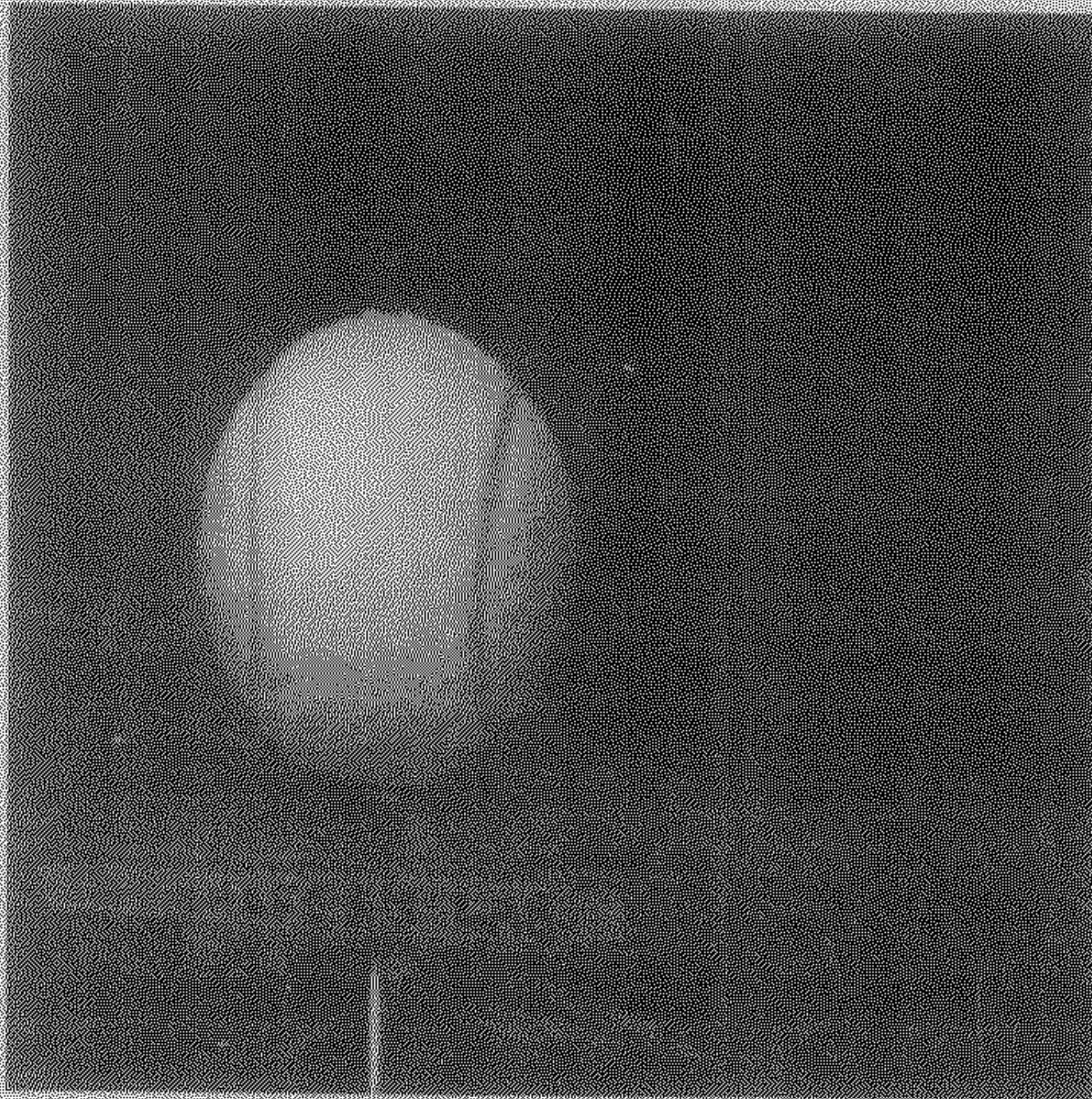
Hand frame 1 - after compression



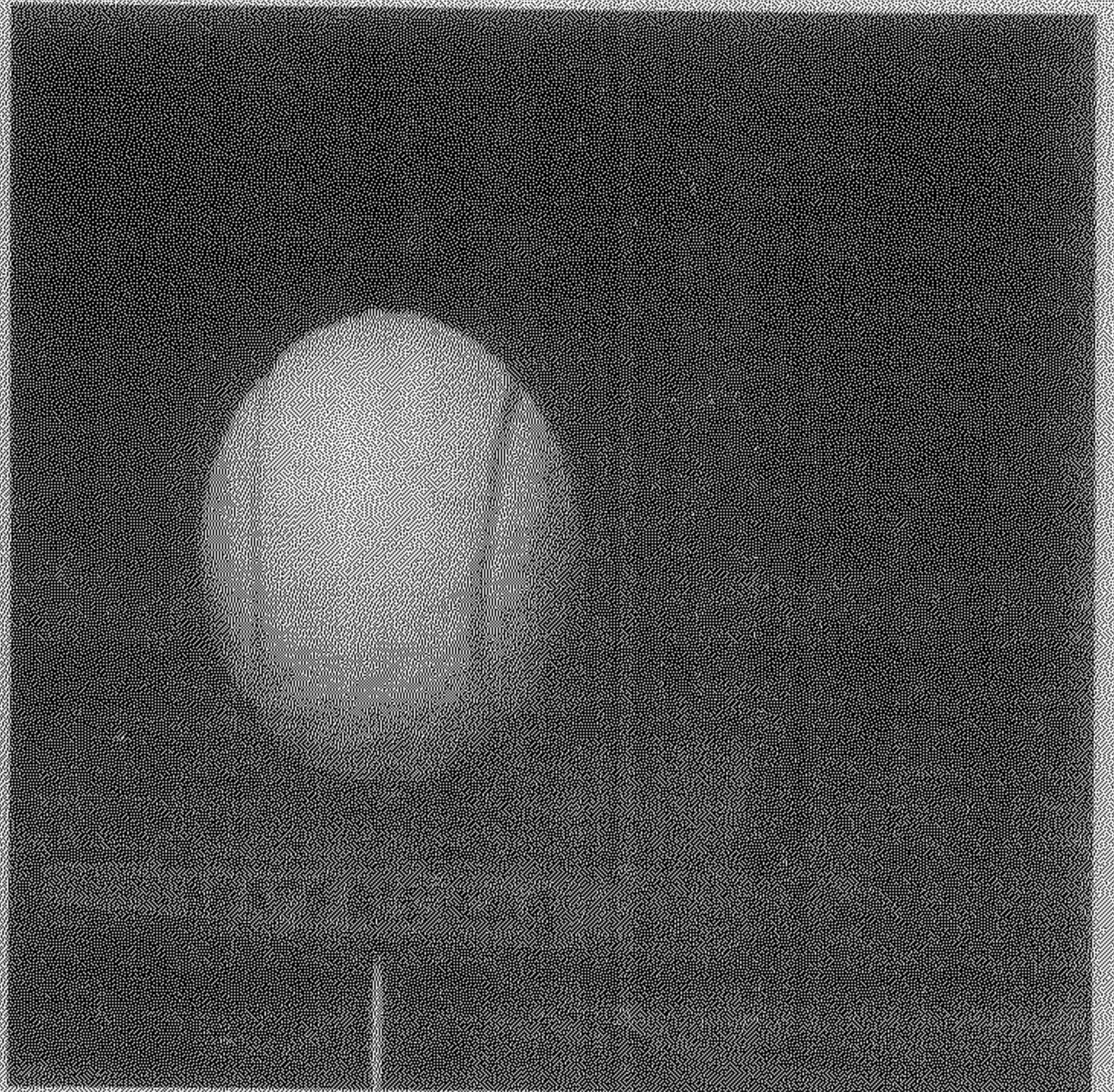
Hand frame 5 - original



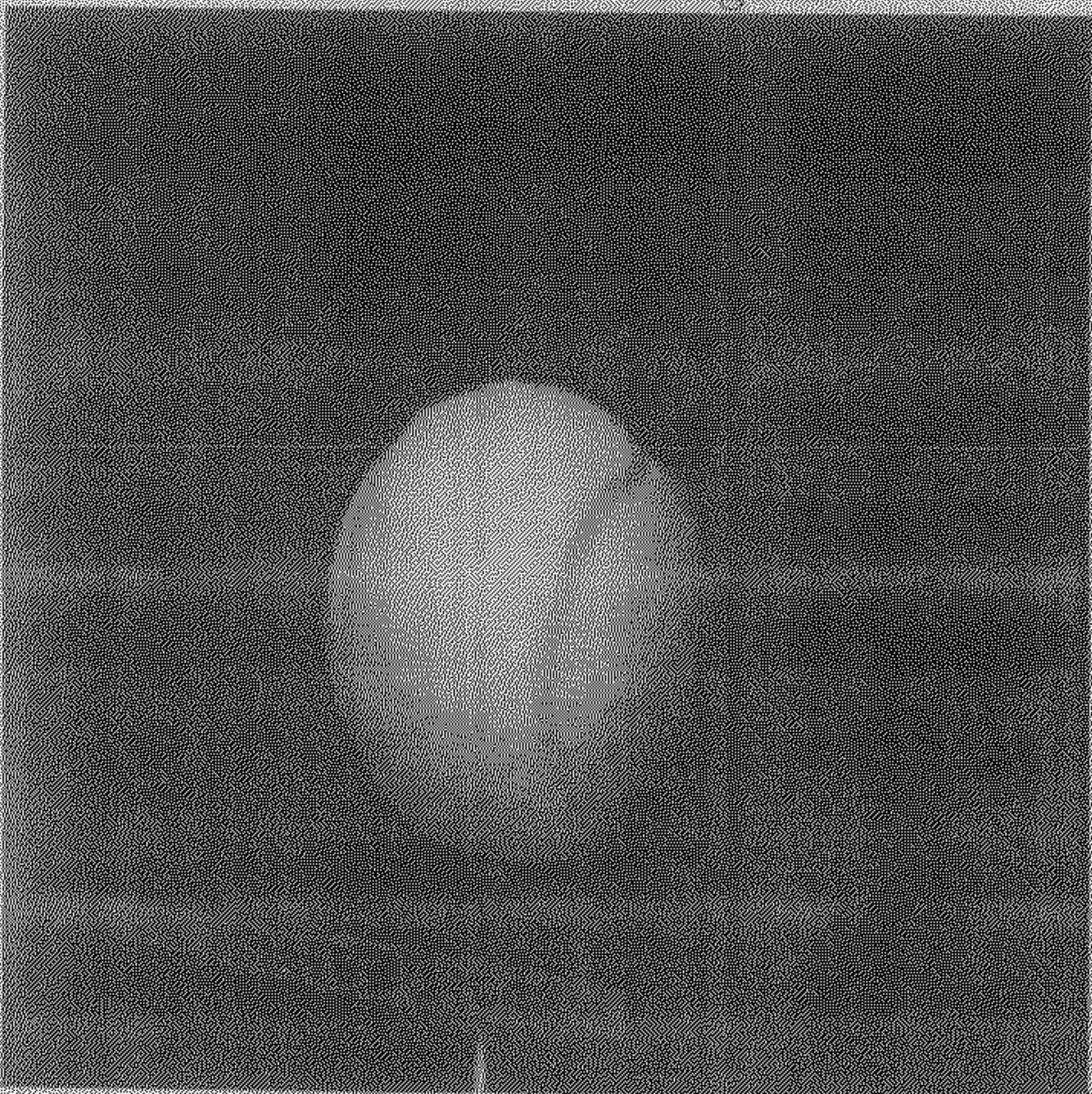
Hand frame 5 - after compression



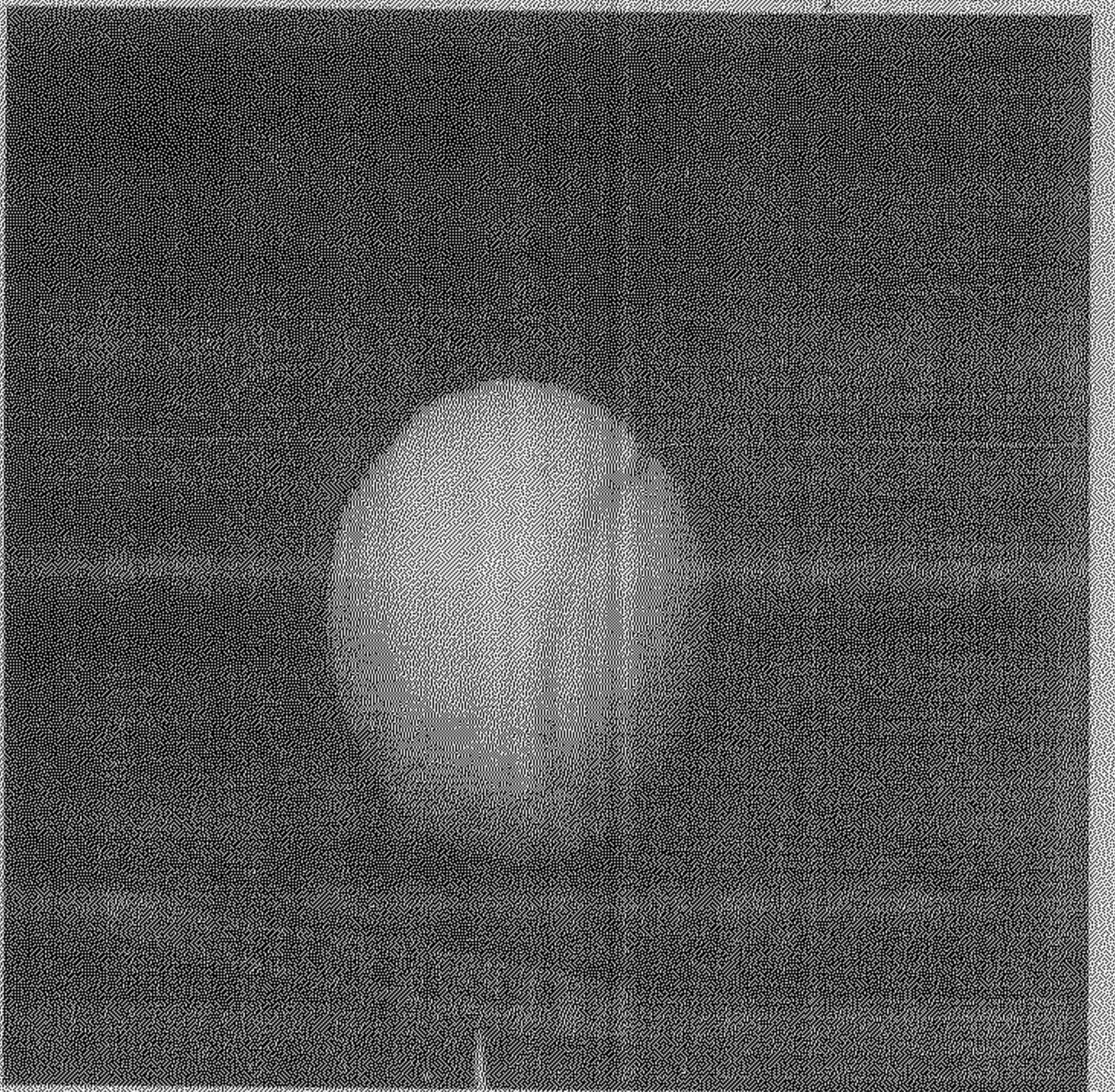
Tennis ball frame 1 - original



Tennis ball frame 1 - after compression



Tennis ball frame 5 - original



Tennis ball frame 5 - after compression

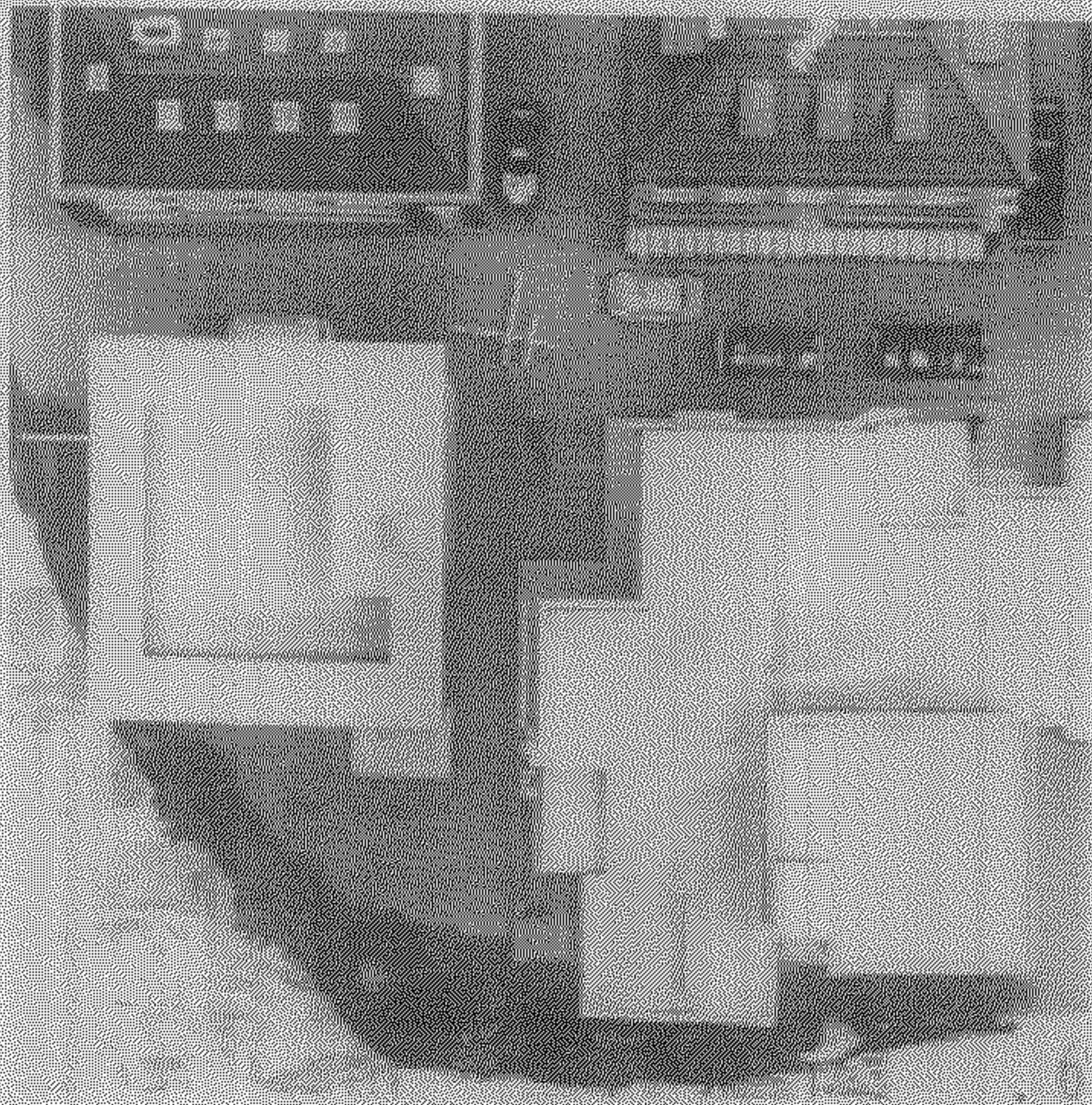
7.0.2 Results of Second method

These are the results of the second method with the datasets mentioned in Section(5.3).

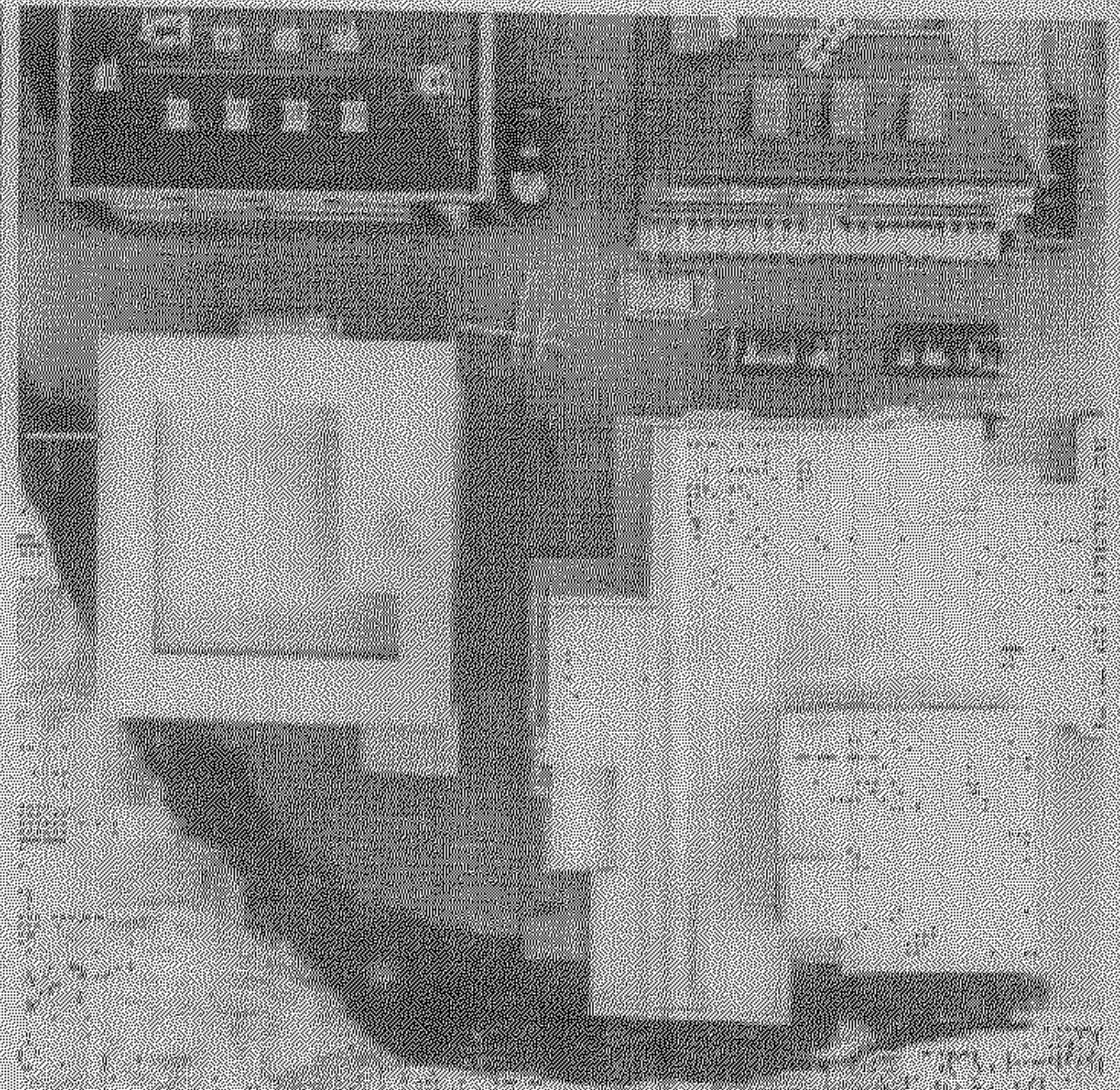
Dataset	PSNR	Compression ratio
LFA	22.14	48:1
Hand	26.54	63:1
Books	24.67	101:1
MIU lab	27.76	102:1
Golf_ball	30.62	162:1

In general the quality of the video should decrease with the increase of compression. But this is not the case with the above table. The reason for this is that the quality of an image is dependent on the number of features present in the image. By observing these images we can find that the LPA sequence has many features compared to the golf sequence hence its quality is less when compared to that of the LPA sequence. Thus if this algorithm is repeated on the same data set by compressing further and further, the quality will as usual decrease further and further.

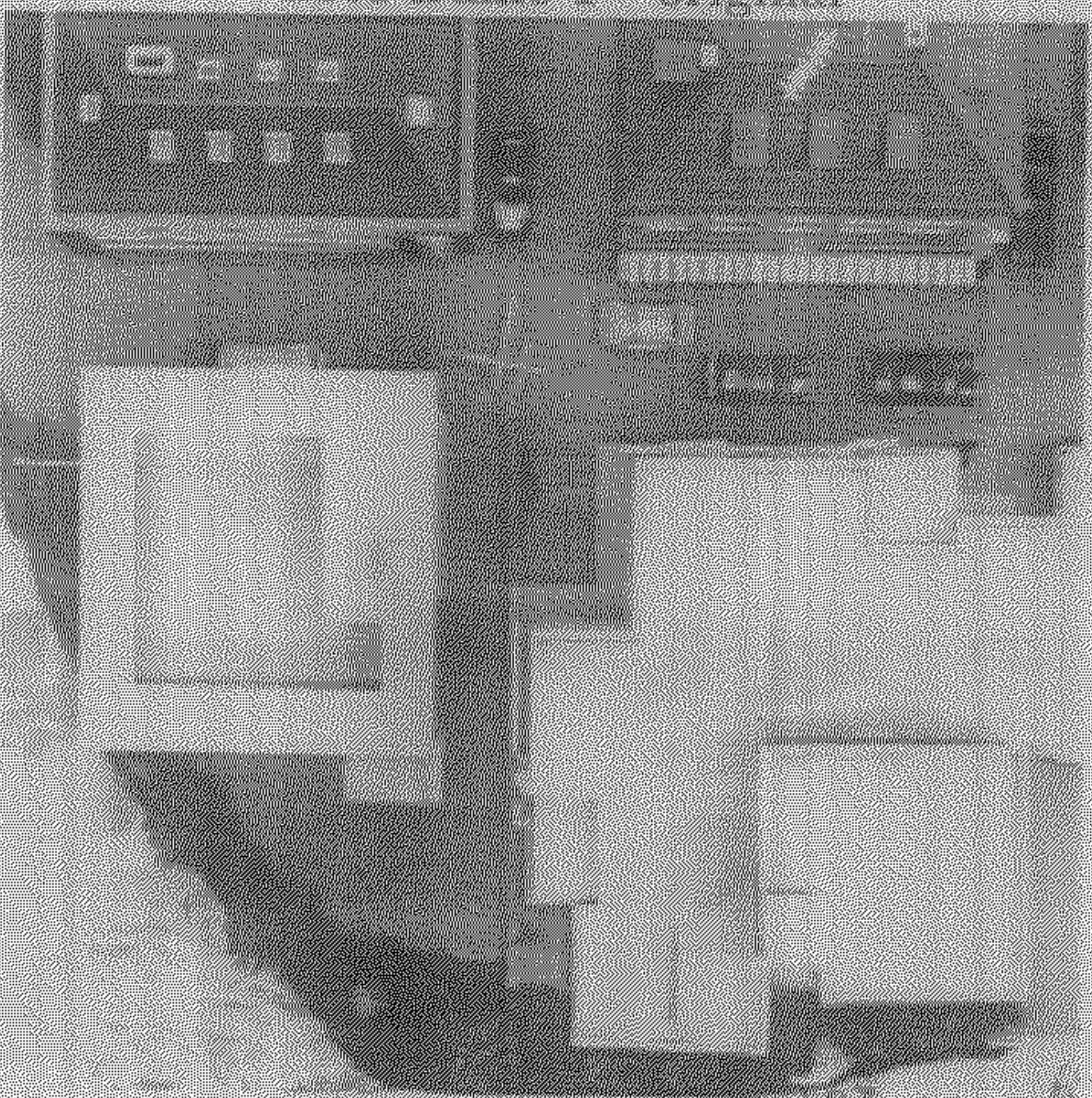
The images obtained after applying the second method on the data sets given in the above table are shown below. In this case the images of the first frame and the sixth frame before compression and after compression are displayed. In this method, for every 10 frames of a dataset, the first frame is coded independently and the frames 2 to 10 are coded using the information about the previous frame.



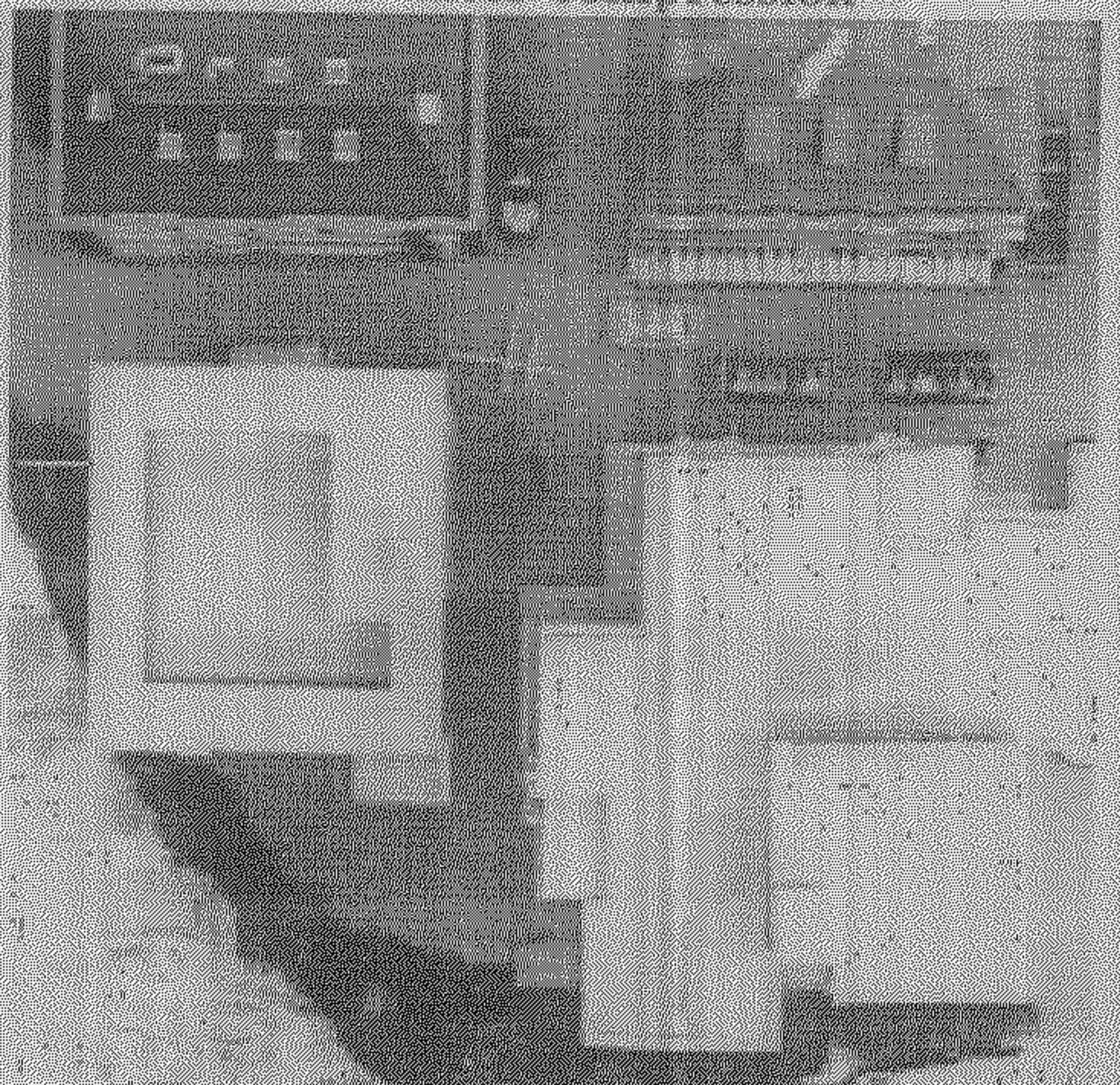
LFA frame 1 - original



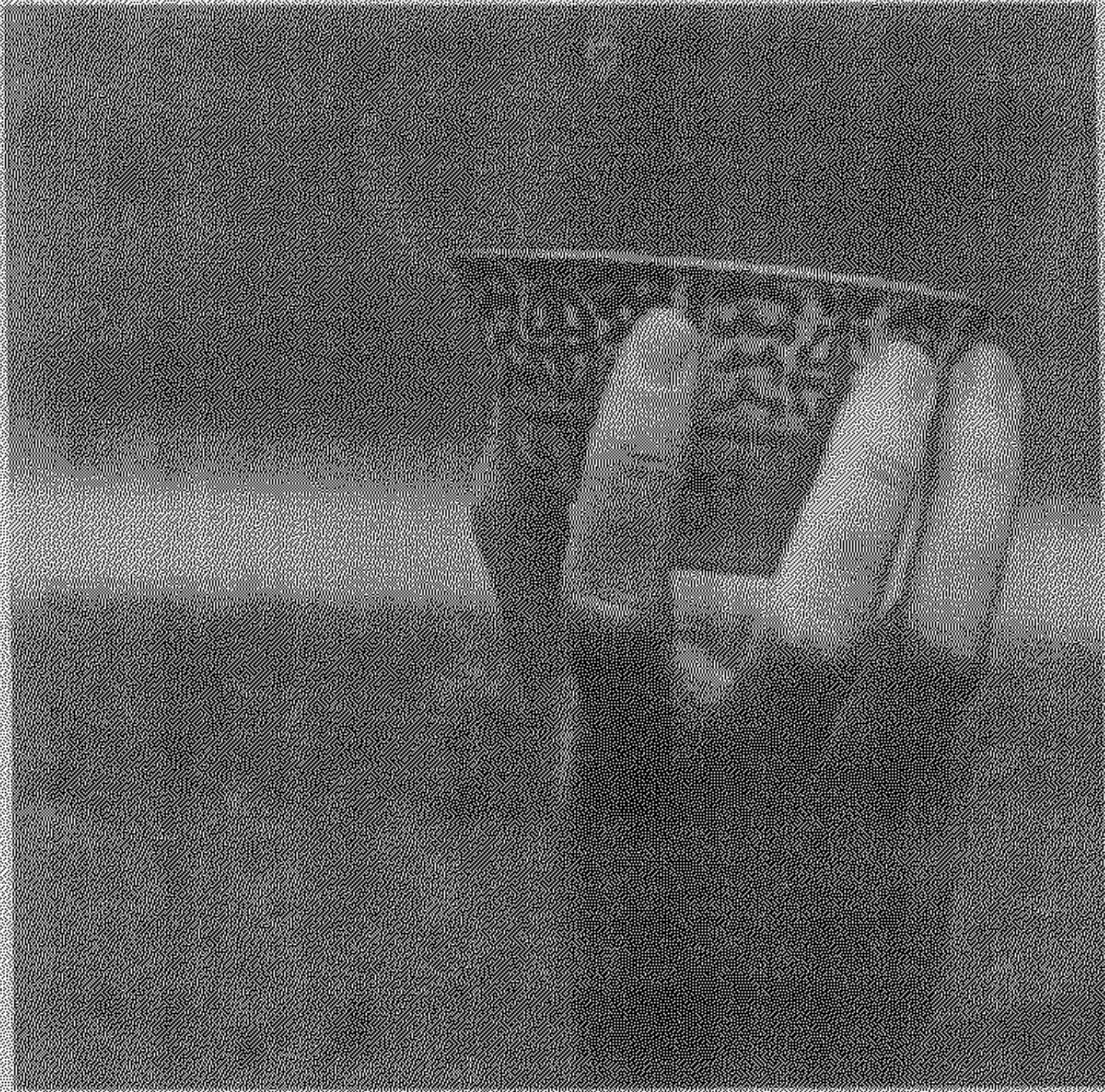
LFA frame 1- after compression



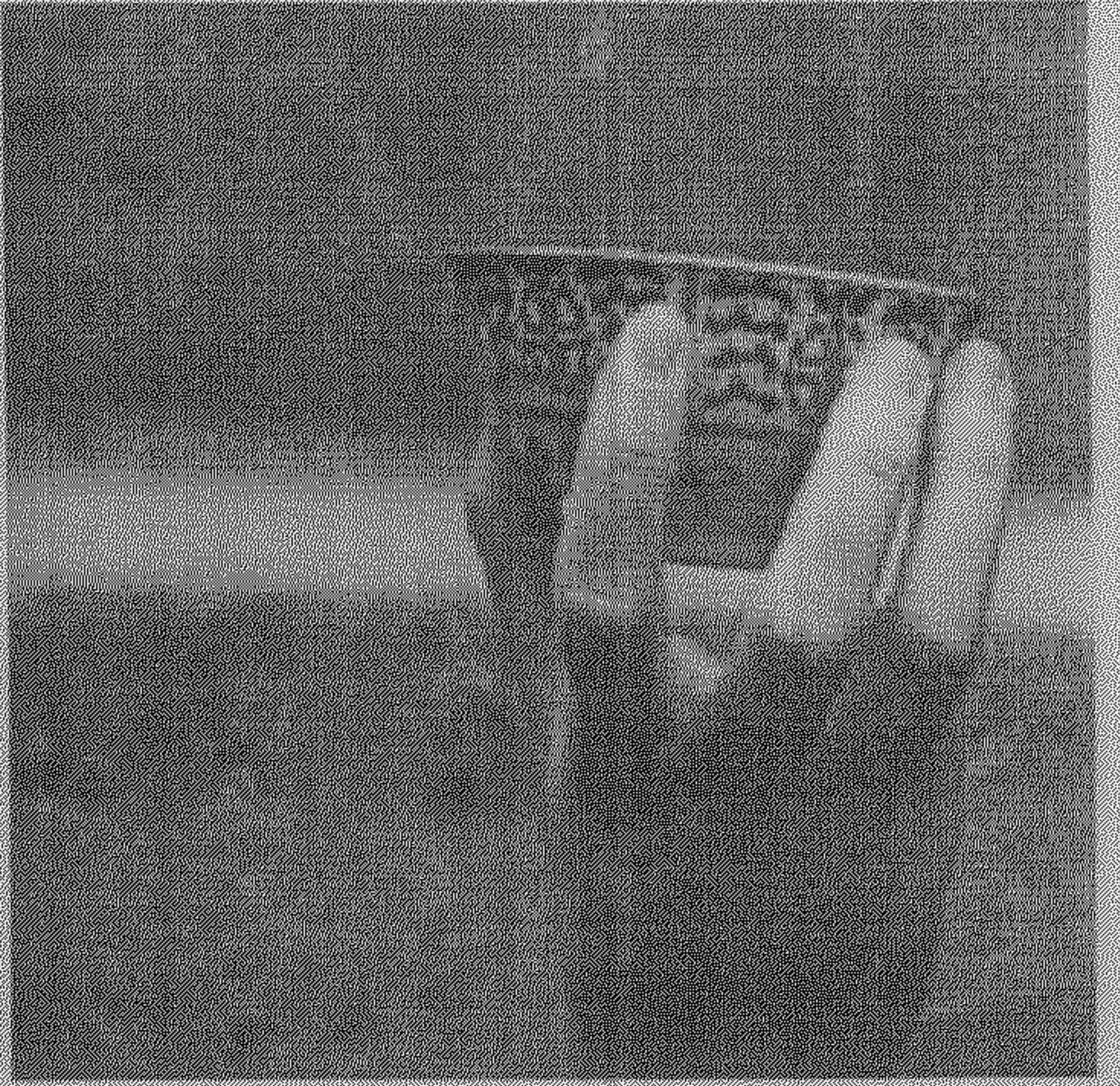
LFA frame 6 - original



LFA frame 6- after compression



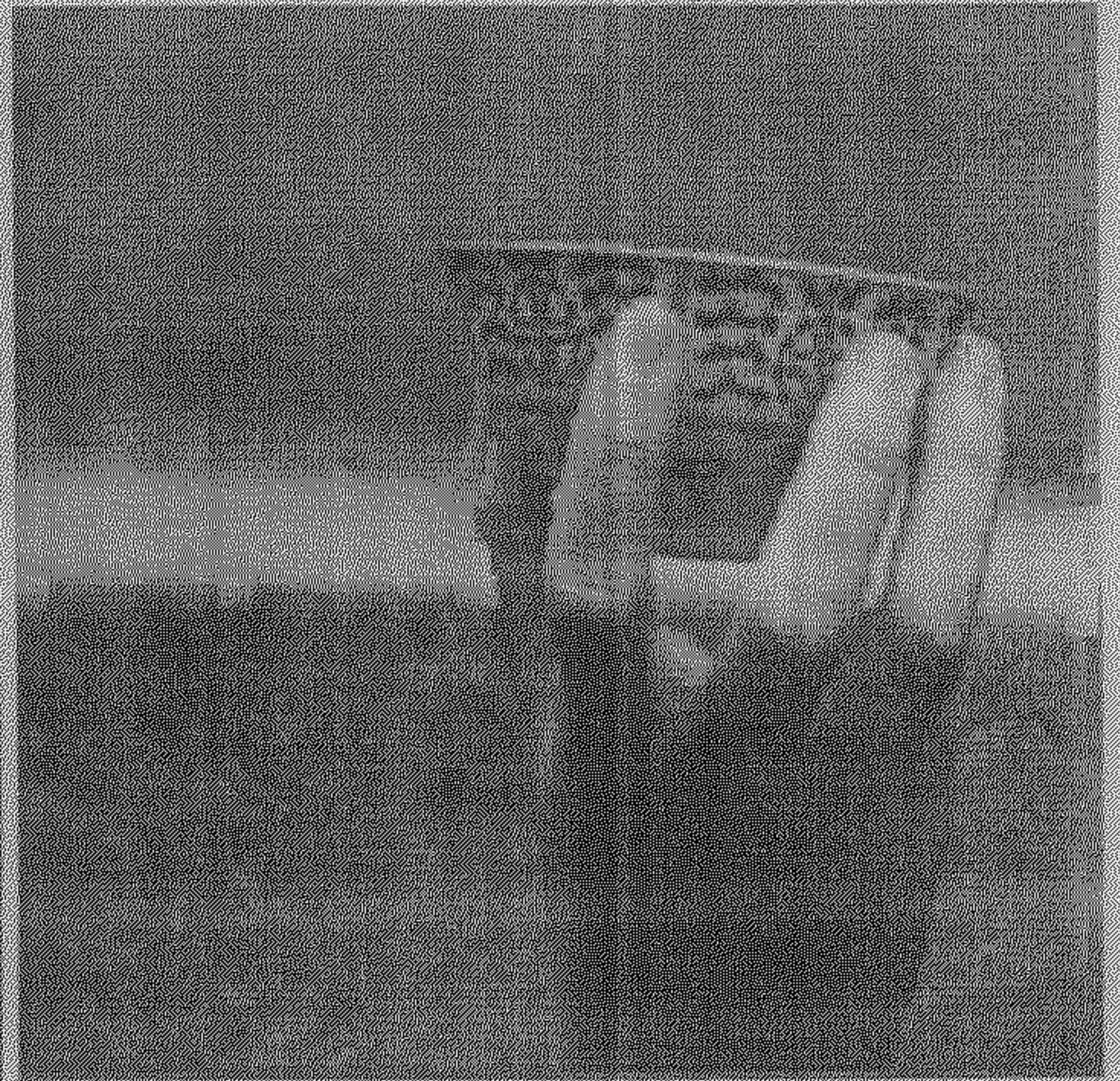
Hand frame 1 - original



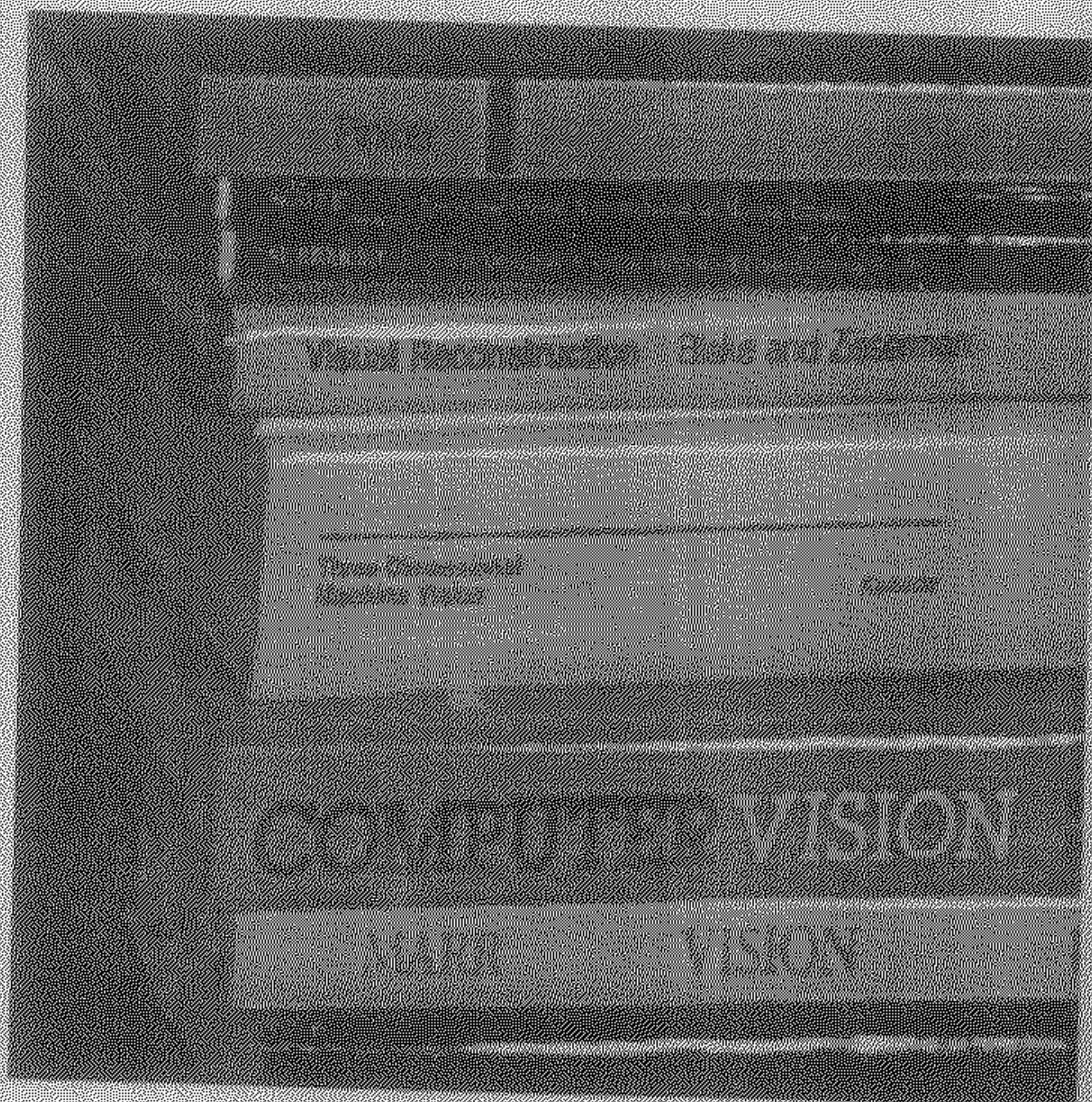
Hand frame 1 - after compression



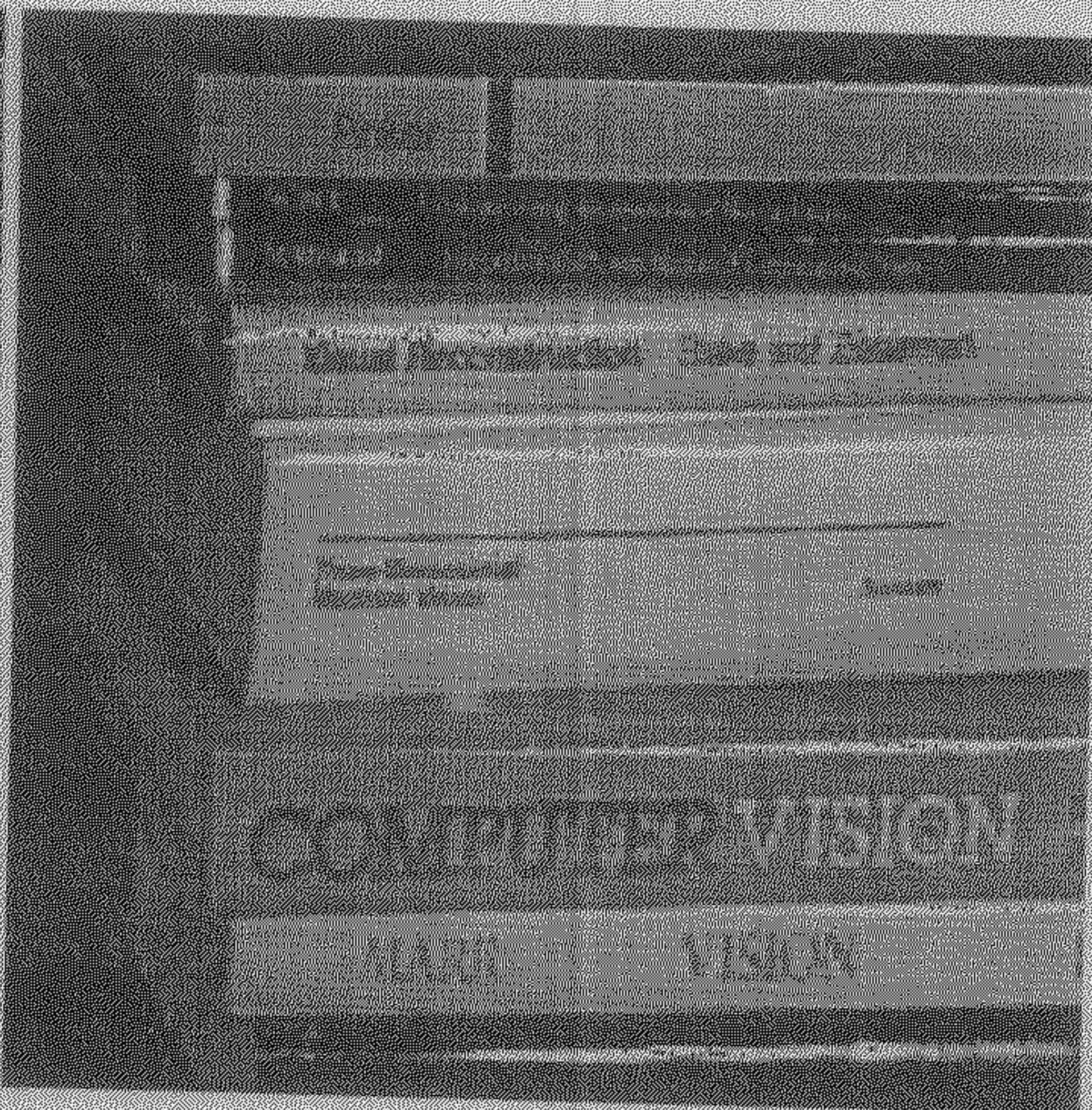
Hand frame 6 - original



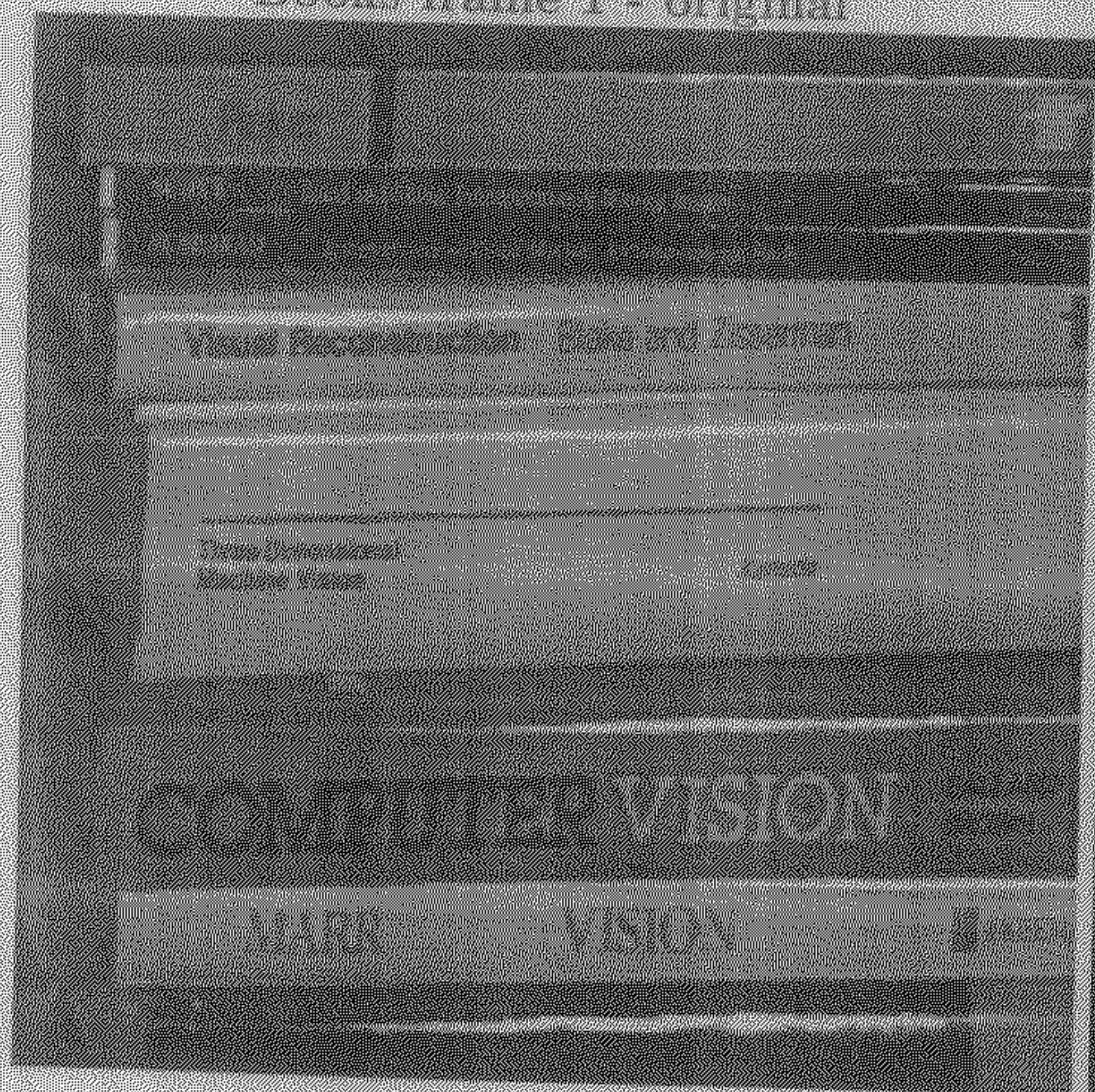
Hand frame 6 - after compression



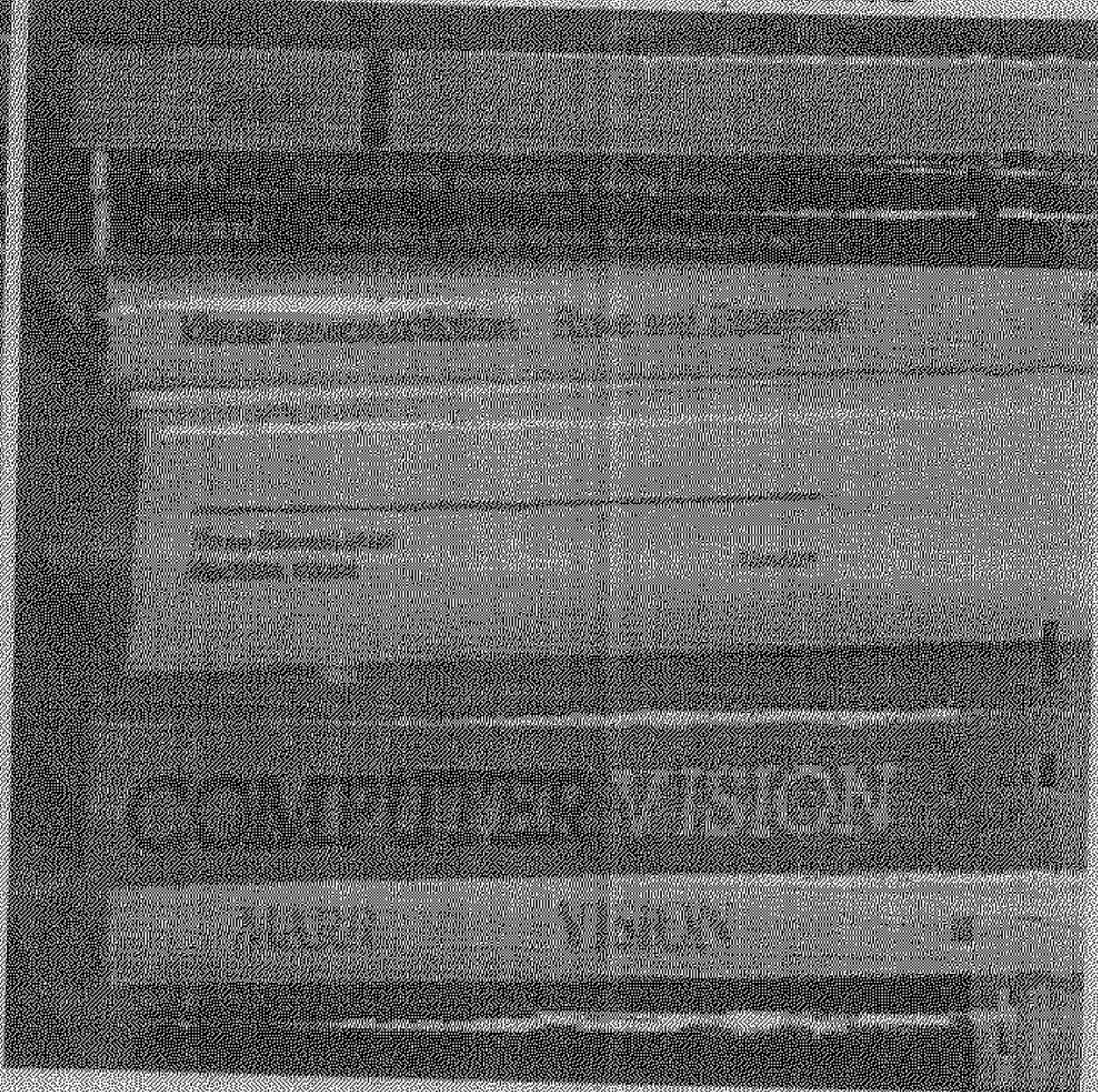
Books frame 1 - original



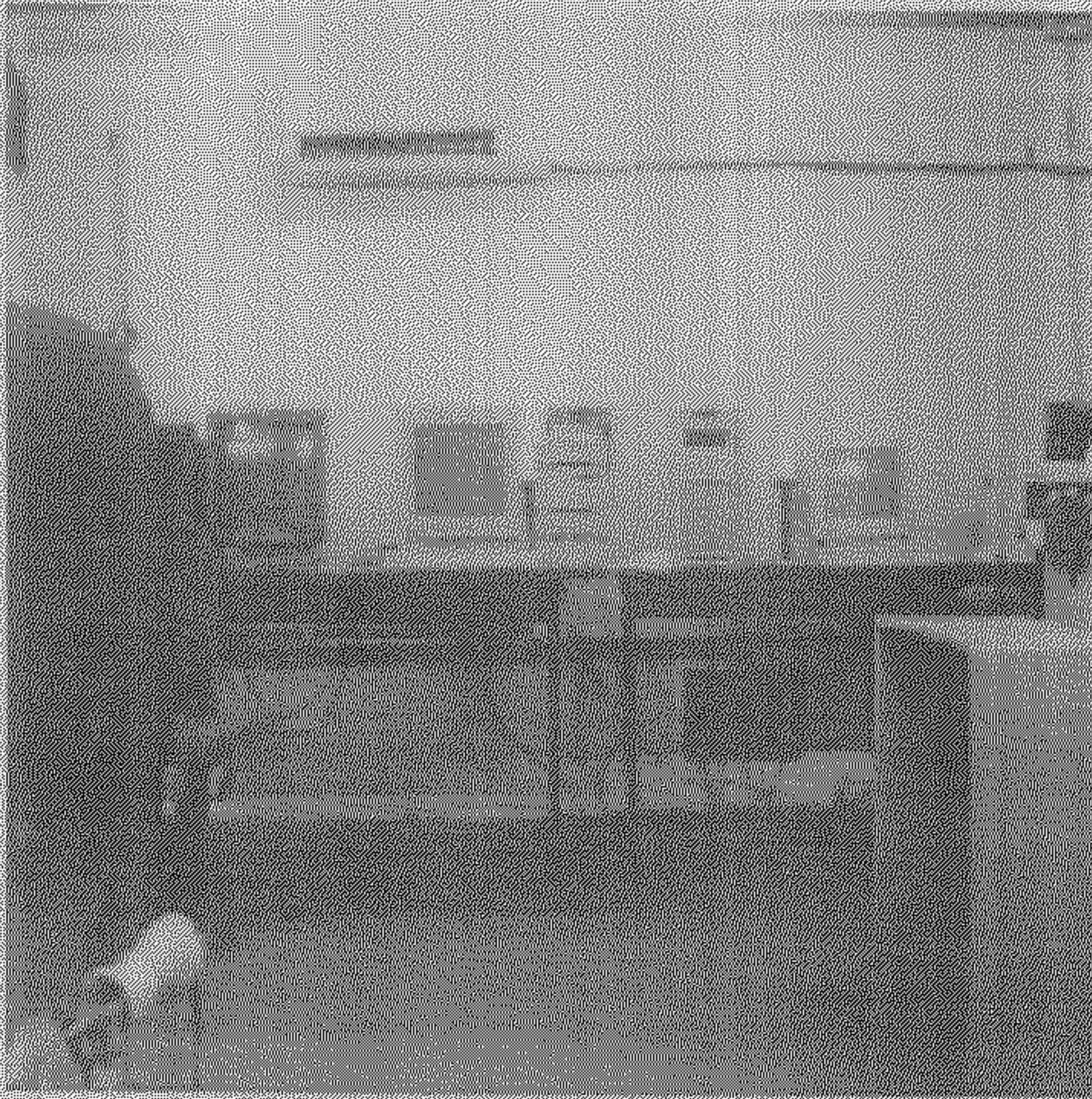
Books frame 1 - after compression



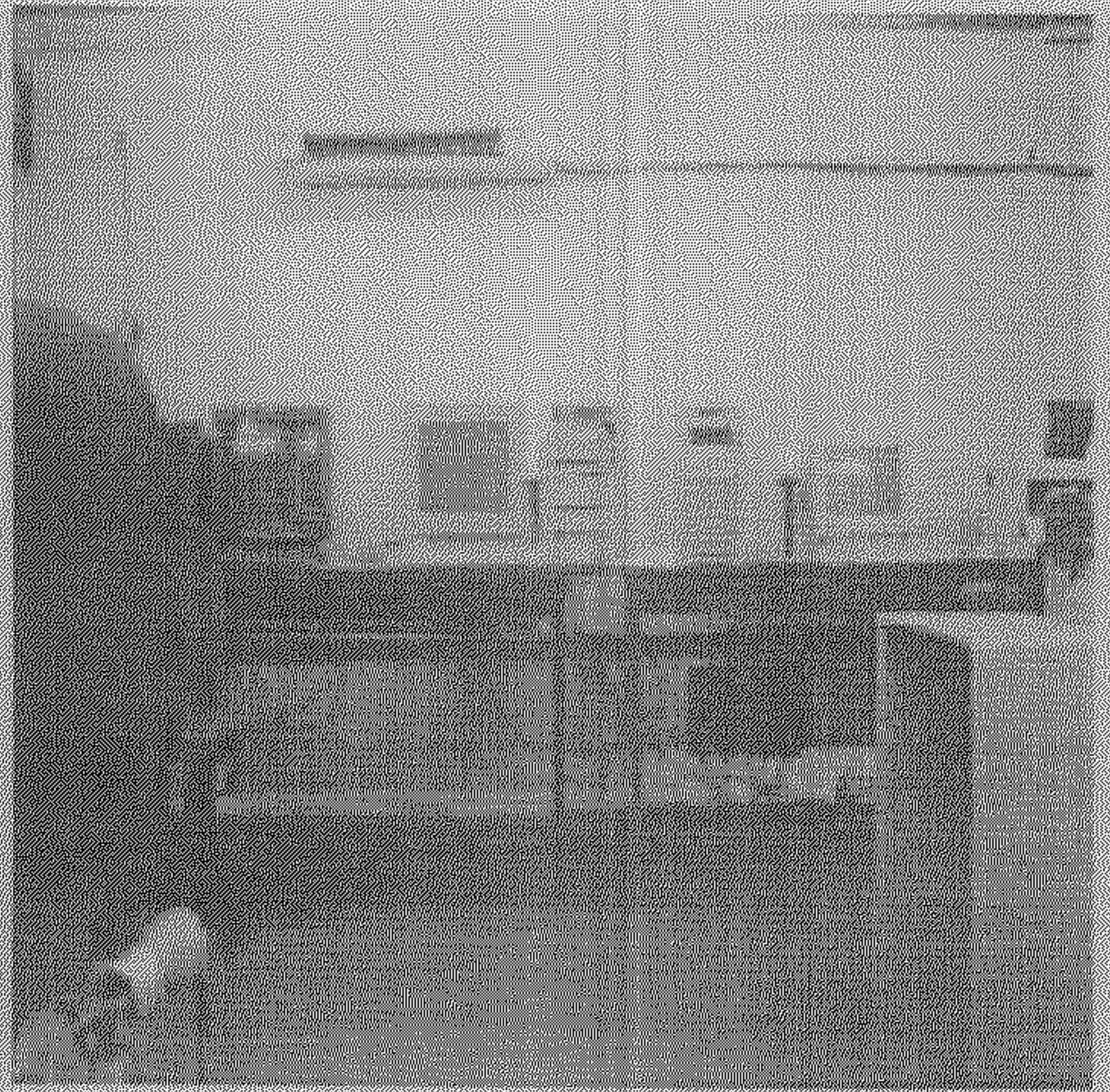
Books frame 6 - original



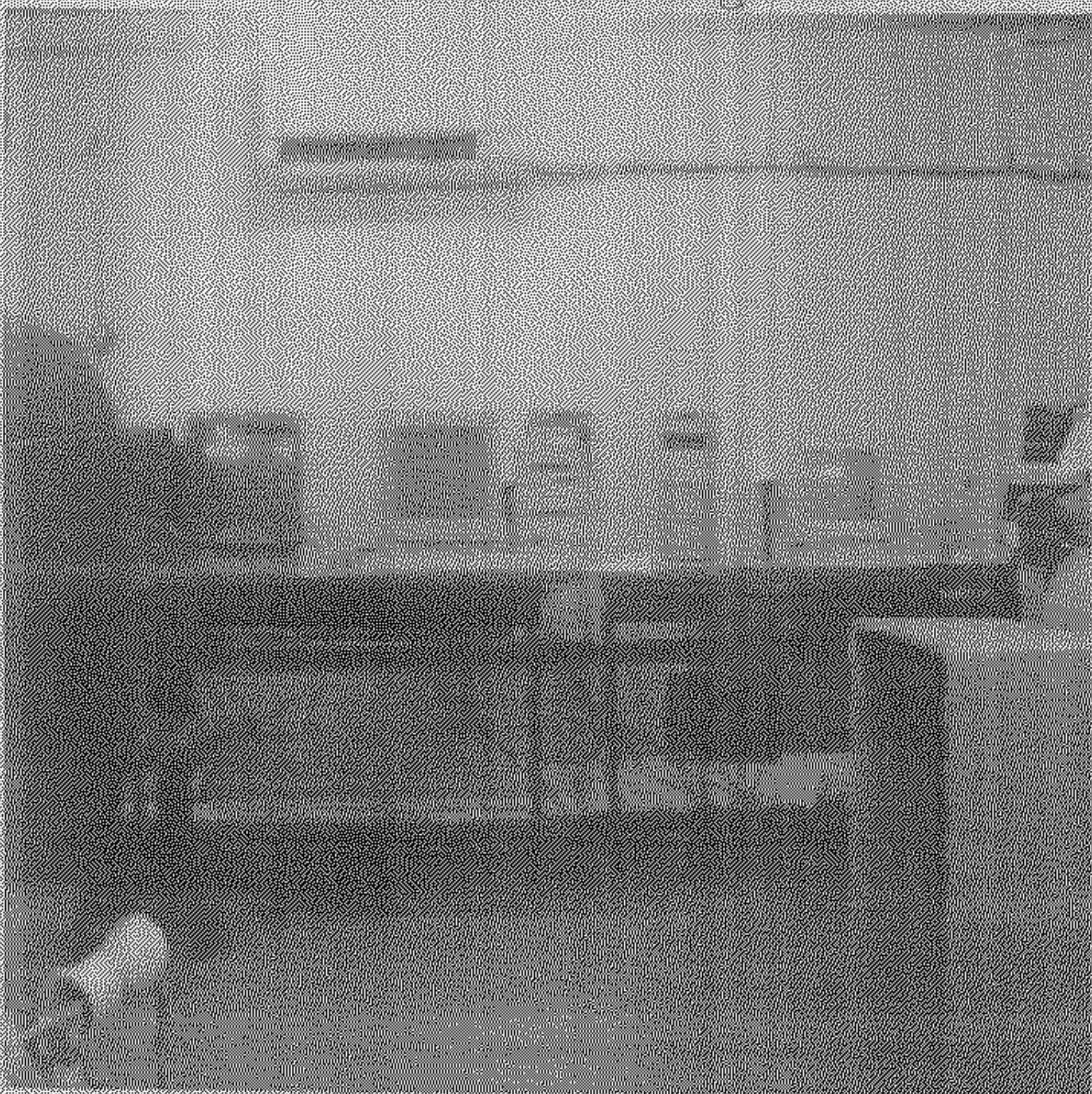
Books frame 6 - after compression



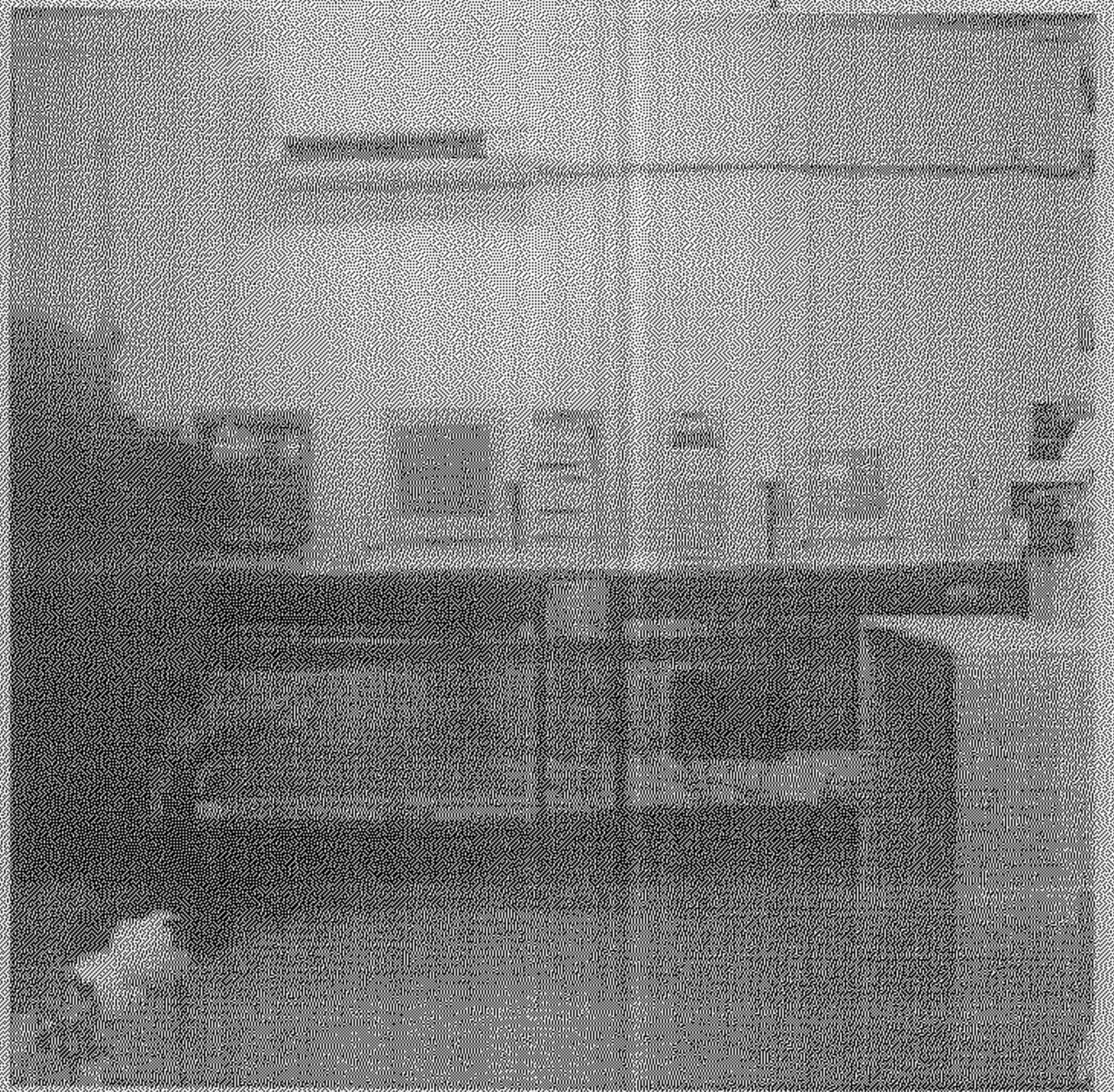
MIU lab frame 1 - original



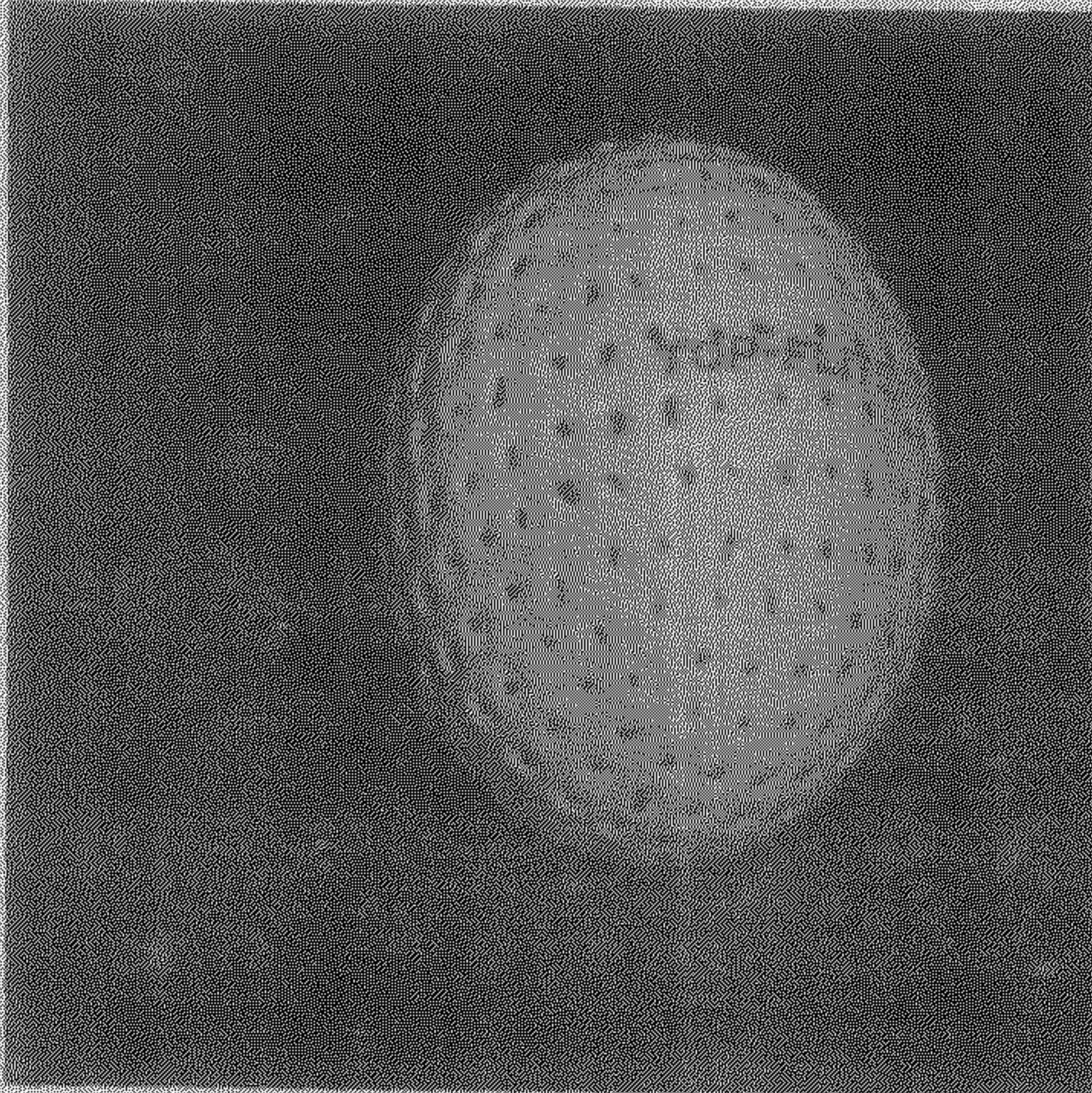
MIU lab frame 1 - after compression



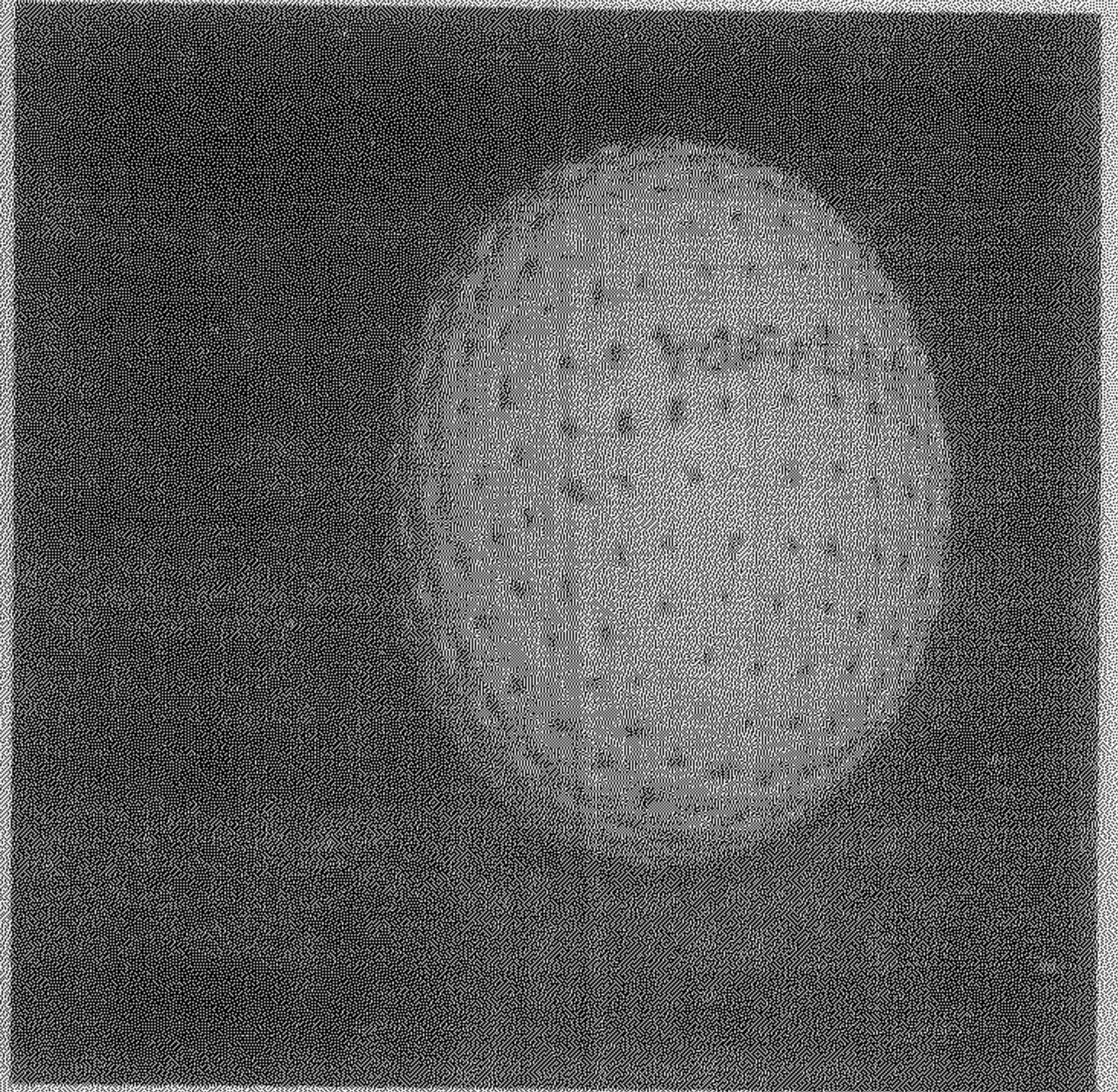
MIU lab frame 6 - original



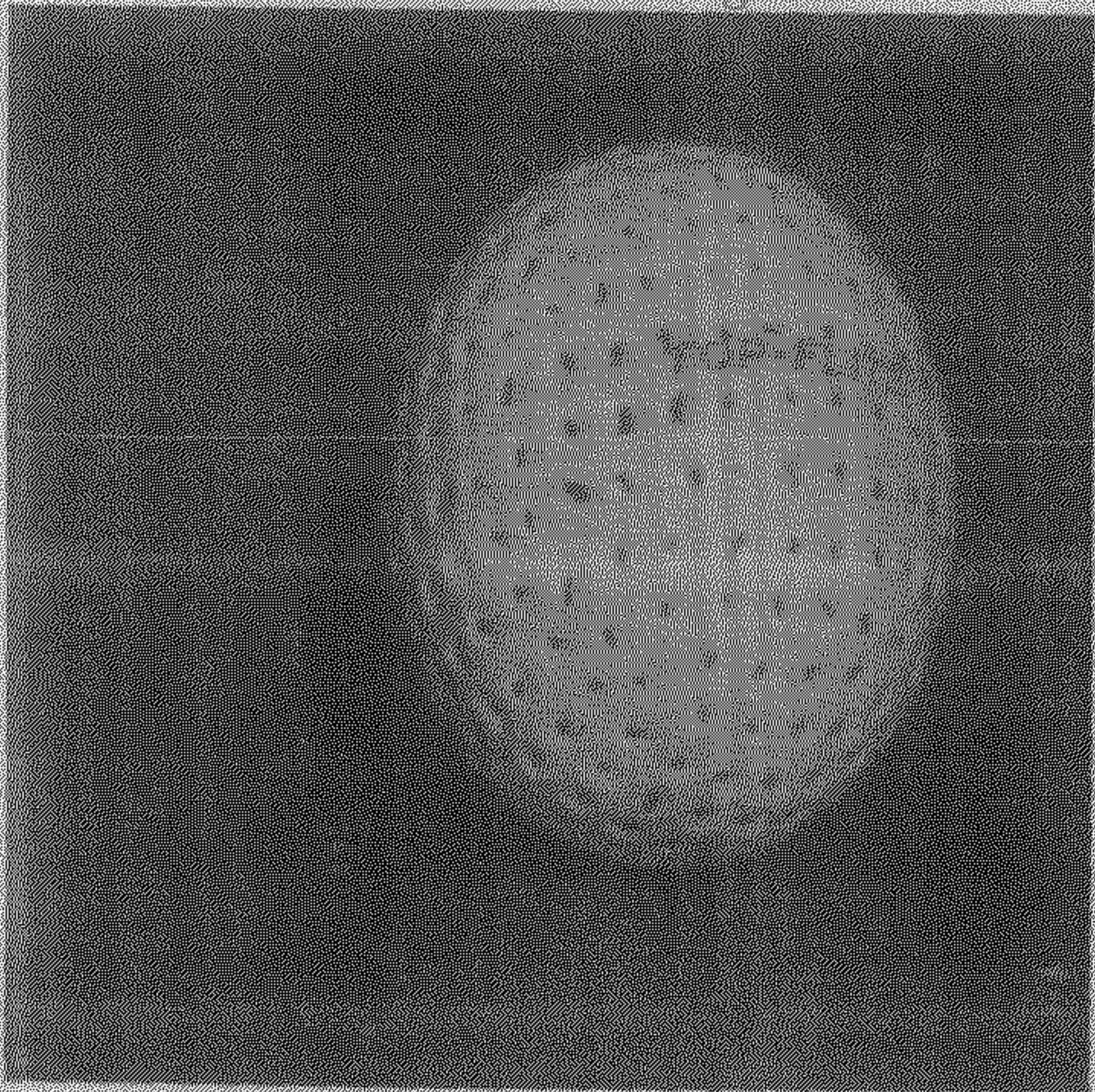
MIU lab frame 6 - after compression



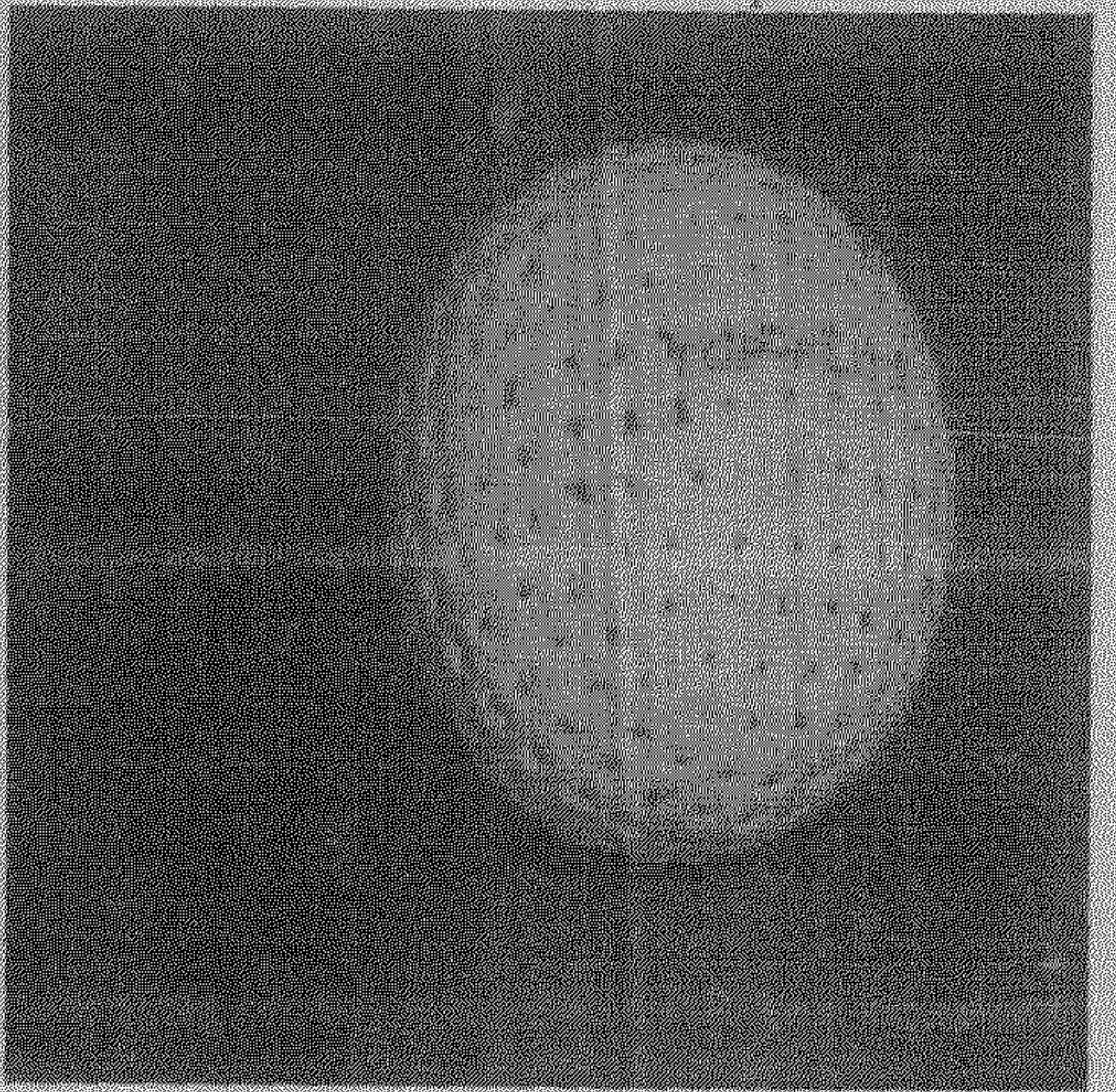
Golfball frame 1 - original



Golfball frame 1 - after compression



Golfball frame 6 - original



Golfball frame 6 - after compression

7.0.3 Comparisons

Compression ratio achieved by MPEG on an SIF¹ video is 26:1 with a good quality. In this project for the purpose of comparison, Miss America video by H.263 is considered. The results of H.263 on Miss America video, a color video with 10 frames per second and with a variable data rate, are shown below.

Bitrate(Kbps)	PSNR	Compression ratio
505.61	48.38	6:1
112.09	43.98	27:1
56.70	41.75	54:1
22.81	38.51	133:1

From the table observe that the maximum compression ratio obtained for this color video is 133:1 where as the maximum compression ratio obtained for gray level Golfball video, using the second method, is 162:1 as shown in the table of section (7.0.2). We know that the color images contain more redundant data when compared to gray level images. So, when the second method will be extended to color images it is expected to give a good compression ratio preserving the quality of the video.

¹SIF is Source Input Format

Chapter 8

Conclusions, discussion and scope for further work

8.1 *Conclusions*

The methods developed in this work are able to provide compression ratio and PSNR values comparable to the standards existing in the world.

It may be noted that the standards are stated for color images not for gray level images. Since the color images generally provide more compression than their gray level counterparts, the methods stated in this dissertation, when generalised to color images, are expected to provide more compression than the existing standards.

8.2 *Discussion and scope for further work*

Though the methods that are suggested are providing good quality video frames and also possess acceptable compression ratio, they have their limitations.

1. They are restricted to gray level images.
2. The first method is not taking advantage of the temporal redundancy.
3. The second method is time consuming and computationally expensive though the standards are also time consuming and expensive.
4. The second method is restricted to type 2 images

Since almost all the video images are now a days colored, the proposed methods should be extended for color images where we can achieve more compression. Instead of just finding the

difference image and coding it, it would be suggestable to use object based modeling techniques where objects can be estimated using the previous image.

Finding an MMV for the current frame can be used to find the MMV for the next frame. That is, instead of searching for a best block in the whole of the frame, it will be computationally easy if we start searching for the best block using the best block information of the previous frame. In this way the algorithm becomes computationally efficient.

The method of finding one MMV for type 2 video can be extended to any type of video by finding MMV's for some big partitions of the current frame where partitions may not be huge continuous blocks. That is, these partitions may contain some gaps which indicate the presence of unmatched blocks. These unmatched blocks can be either estimated using object model estimation [1] or by some coding techniques.

Though these methods are implemented using Haar wavelets, they can be done using any wavelet basis. So one can develop one's own wavelets and find the best one which is suitable for the video sequence in hand to attain much more compression. In these methods wavelet transformation is applied at only one level, if it is applied for more and more levels and different portions are quantized using different quantization techniques then quality of the images can be further improved. Alternating schemes like lifting scheme can be adopted to speed-up when compared to the standard implementation. Lifting allows for an in-place implementation of the fast wavelet transform, a feature similar to the fast fourier transform[18].

In all the data sets used in the second method much of the information is lost while coding an intra frame. This information is mainly lost in the quantization step. Since the next inter frame is coded using the intra frame, its quality is deteriorated and it is carried on for all the subsequent inter frames. Thus if we can find a suitable quantization technique for the intra frame then a good quality video can be obtained maintaining a good compression ratio.

The computer based methods of recent times are generally interactive. User will be satisfied if he can specify either the compression ratio or the quality of the video and gets the video accordingly. Research work is going on in this area. These algorithms that are developed can also be made interactive by considering different levels of wavelet transform and performing the quantization according to the bit rate specified by the user. The lower the bit rate, the coarser the wavelet coefficients have to be quantized. This is a matter for further studies.

Bibliography

- [1] A. Murat Tekalp, *Digital Video Processing*, Prentice Hall, 1995.
- [2] David Salomon, *Data Compression - The Complete Reference*, second edition, Springer-Verlag Newyork, Inc., 2000..
- [3] C.Sidney Burrus, Ramesh A.Gopinath and Haitao Guo, *Introduction to Wavelets and Wavelet Transforms*, Prentice Hall, New Jersey, 1998.
- [4] Tom H. Koornwinder, *Wavelets - An elementary treatment of theory and applications*, World Scientific, Singapore, 1993.
- [5] C. Valens, "A really friendly guide to Wavelets", 1999, clements@polyvalens.com.
- [6] Daubechies, I., *Ten Lectures on Wavelets*, SIAM, Philadelphia, 1992.
- [7] Stephane Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, 1999.
- [8] Rafael C.Gonzalez and Richard E.Woods, *Digital Image Processing*, Addison Welsey Longman Singapore pte. Ltd., 1999.
- [9] A. Rosenfeld and A. Kak, *Digital Image Processing*, Academic Press, 1972.
- [10] Mallat H., "A theory for multiresolution signal decomposition : the wavelet representation", *IEEE Transactions in Pattern Analysis and Machine Intelligence*, vol. 11, pp. 674-692, 1989.
- [11] Ingrid Daubechies, "Orthonormal bases of compactly supported wavelets", *Comm. Pure and Applied Math.*, vol 41, pp 909-996, 1988.
- [12] Diego Santa-Cruz and Touradj Ebrahimi, "An analytical study of JPEG 2000 Functionalities", *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, vol. 2, pp. 49-52, September, 2000.

- [13] Guy Cote, Michael Gallant, and Faouzi Kossentini, "Efficient Motion Vector Estimation and Coding for H.263-Based Very Low Bit Rate Video Compression", *IEEE transactions on circuit and systems for video technology*, vol. 8, no. 7, pp. 849, November 1998.
- [14] Christopher Lau, James E. Cabral, Jr., Avni H. Rambhia, and Yongmin Kim, "MPEG-4 Coding of Ultrasound Sequences", *SPIE, Medical Imaging 2000 Display conference*, vol. 3976, pp. 573-579, 2000.
- [15] "Video Compression - An Introduction", Array Microsystems, Inc, 1997, <http://www.array.com/compress.pdf>.
- [16] Francescomaria Marino, Tinku Acharya, Lina J. Karam, "A DWT-Based Perceptually Lossless Compression Scheme and VLSI Architecture Suitable for R-G-B Digital Images".
- [17] Brian Schoner, John Villasenor, Steve Molloy and Rajeev Jai, "Techniques for FPGA Implementation of Video Compression Systems", *FPGA*, pp. 154-159, 1995, citeseer.nj.nec.com/23114.html.
- [18] Ingrid Daubechies and Wim Sweldens, "Factoring Wavelet transforms into lifting steps", September 1996, bellabs.com/cm/ms/who/...factor.ps.gz.