

DIGITAL WATERMARKING



159159

FIELD OF THE INVENTION

The present invention relates to digital watermarking, and in particular to a method and
5 system for use in watermarking image data, and a method and system for detecting a
watermark.



G00002

BACKGROUND

Digital watermarking is a technique used to hide digital data – the digital watermark – in
10 digital multimedia data. Digital watermarking is used for copy control, data labelling,
copyright protection, pirate tracing, and other digital rights management functions. Digital
watermarks can be categorised as fragile or robust. A fragile watermark is a watermark
that is easily affected by changes made to the watermarked digital data. Fragile
watermarks are therefore useful for detecting whether the watermarked data has been
15 modified. Conversely, robust watermarks are designed to resist any tampering to the
watermarked data. A robust watermark is embedded into perceptually significant
multimedia content to ensure that it remains with the data even if the data is modified.

Digital multimedia exists in an enormous variety of formats. Two of the most ubiquitous
20 image formats are the JPEG format for still images, and the MPEG format for moving
image or video data. The JPEG and MPEG encoding methods translate image (still and
video, respectively) data from the spatial domain to the frequency domain by a method
known as discrete cosine transform (DCT). The frequency domain data, a set of DCT
coefficients, is then compressed by quantizing the DCT coefficients by different amounts
25 according to frequency, allowing substantial overall compression without significantly
degrading the perceived image quality. For example, DCT coefficients corresponding to
higher frequencies are generally compressed more than low frequency coefficients.

As shown in Figure 1, an MPEG/JPEG encoder 10 includes a discrete cosine transform (DCT) module 12, a quantization module 14, and a variable length entropy coding (VLC) module 16. In the case of MPEG video, the spatial image data 18 can be further compressed by a technique called motion compensation whereby subregions of a frame are generated from corresponding subregions of another frame whose location in the frame may be different due to motion. In either cases, MPEG/JPEG image data is translated from the spatial domain into the frequency domain by the DCT module 12 to produce an 8x8 matrix or block of DCT coefficients. These coefficients are then selectively quantized by the quantization module 14 using an 8x8 matrix of quantization coefficients. The resulting quantized coefficients are variable length entropy coded by the VLC module 16 to produce an MPEG/JPEG bit-stream 20 of compressed data. To view the image or video data, an MPEG/JPEG decoder 22 is used to reverse the operations performed by the encoder 10. That is, an inverse VLC module 24 removes the variable length coding from the MPEG/JPEG bit-stream 20 to produce matrices of quantized DCT coefficients. An inverse quantization module 26 then removes the quantization to produce an 8x8 matrix of DCT coefficients in the frequency domain. These coefficients are then processed by an inverse DCT module 28 to produce spatial video data 30. However, because the encoding process is lossy due to quantization and, in the case of MPEG video, motion compensation, the resulting decoded spatial image data 30 is not identical to the original input image data 18.

Transaction or user specific information can be embedded in still or video image data to enable content distributors to trace copyright violators in cases of suspected piracy. For many applications, such as video-on-demand and video streaming, the source image data is already stored in a compressed format. For applications of this nature, very fast watermarking or fingerprinting of compressed still or video image data is required. Existing watermarking systems are generally not designed for handling compressed data, and are too slow for real-time embedding of watermark data into MPEG/JPEG bitstreams.

Prior art watermarking systems have been used to embed watermark data at various stages of the encoding process, as shown in Figure 2. A typical watermarking system embeds

watermark data in the spatial domain. When used to add watermark data to an existing MPEG/JPEG bitstream 20, the bitstream 20 must be completely decoded by the decoding modules 24 to 28 of the decoder 22 to generate decoded image data 30. A spatial watermarking module 32 then embeds spatial watermark data into the decoded image data 30, and the resulting watermarked spatial image data is encoded by the standard encoding modules 12 to 16 of an encoder 10 to generate a watermarked MPEG/JPEG bitstream 38.

Alternatively, a more efficient watermarking system has been developed to perform watermarking in the DCT (frequency) domain. This requires only partial decoding using the inverse VLC module 24 and the inverse quantization module 26 to generate DCT image data. DCT watermark data is then embedded in the DCT image data by a DCT watermarking module 34 to generate watermarked DCT image data that is then quantized by the quantization module 14 and VLC coded by the VLC module 16 to generate the watermarked MPEG/JPEG bitstream 38. These two watermarking systems require computationally intensive decompression and recompression steps. In the case of MPEG video data, the computational load is exacerbated if motion-compensation is performed in the re-compression process. Consequently, neither system is expected to be suitable for real-time watermarking applications. These prior art schemes also do not specifically consider how to reduce visual distortions introduced by the watermark in an efficient manner, *i.e.* without slowing down the embedding process excessively.

Recently, watermarking in the bitstream domain has been proposed, as described in A. Hanjalic *et. al.*, *Image And Video Databases: Restoration, Watermarking And Retrieval*, New York, Elsevier Science, 2000. As shown in Figure 2, a VLC watermarking module 36 embeds watermark data at the bit-stream level using selected VLC codewords that do not differ significantly in length from the unwatermarked codewords. However, although such an approach is computationally efficient, the authors admit that this approach can be easily attacked by rewatermarking or by arbitrarily changing the limited set of VLC codewords used.

It is desired, therefore, to provide a method and system for use in watermarking image data, and a method and system for detecting a watermark, that alleviate one or more of the above difficulties, or at least provide a useful alternative.

5 SUMMARY OF THE INVENTION

In accordance with the present invention there is provided a method for use in watermarking image data, including generating watermarked quantized image data by linearly adding watermark data to quantized image data.

- 10 The present invention also provides watermarked image data generated by executing the above method.

The present invention also provides a computer readable storage medium having stored thereon watermarked image data generated by executing the above methods.

15

The present invention also provides a method for detecting a watermark in watermarked image data generated by linearly adding watermark data to quantized image data, including correlating the watermarked image data with a predetermined spreading code.

- 20 The present invention also provides a method for detecting a watermark in watermarked image data generated by linearly adding watermark data to quantized image data, including inverse quantizing said watermarked image data, and correlating the inverse quantized watermarked image data with a spreading code.

- 25 The present invention also provides a method for detecting a watermark in watermarked image data generated by linearly adding watermark data to quantized image data, including inverse quantizing said watermarked image data and a spreading code, and correlating the inverse quantized watermarked image data with the inverse quantized spreading code.

The present invention also provides a method for detecting a watermark in watermarked image data generated by linearly adding watermark data to quantized image data, including inverse quantizing and inverse transforming the watermarked image data and a spreading code, and correlating the inverse quantized and inverse transformed watermarked image data with the sign of an inverse quantized and inverse transformed spreading code, said spreading code comprising a sequence of two values having opposite sign..

The present invention also provides a method for detecting a watermark in watermarked image data generated by linearly adding watermark data to prediction reference frames of quantized image data and drift compensation data to predicted frames of said quantized image data, said method including performing correlation on the reference frames to generate reference correlator output, performing correlation on predicted watermark data corresponding to said predicted frames to generate predicted correlator output, subtracting the predicted correlator output from the reference correlator output, and thresholding the result.

The present invention also provides a method for detecting a watermark in watermarked image data generated by linearly adding watermark data to prediction reference frames of quantized image data, and drift compensation data to predicted frames of said quantized image data, including generating a despreading matrix for a predicted block of image data from despreading matrices for up to four corresponding reference blocks of the watermarked image data.

The present invention also provides a method for detecting a watermark in watermarked image data generated by linearly adding watermark data to quantized image data, including selecting quantized transform coefficients of said watermarked image data, and correlating the selected coefficients with a predetermined spreading code.

The present invention also provides a system having components for executing the steps of any one of the above methods.

The present invention also provides a computer readable storage medium having program code stored thereon for executing the steps of any one of the above methods.

- The present invention also provides a watermarking system, including a module for
5 generating watermarked quantized image data by linearly adding watermark data to quantized image data.

The present invention also provides a system for generating watermarked quantized image data, including a module for selecting portions of image data on the basis of a human
10 visual system analysis of the image data, and a module for embedding a watermark in the selected portions of the image data by linearly adding watermark data to quantized image data.

The present invention also provides a system for detecting a watermark in watermarked
15 quantized image data generated by linearly adding watermark data to quantized image data, said system including a detection module for detecting said watermark in a quantized domain by correlation and thresholding.

BRIEF DESCRIPTION OF THE DRAWINGS

20 Preferred embodiments of the present invention are hereinafter described, by way of example only, with reference to the accompanying drawings, wherein:

Figure 1 is a block diagram of a prior art MPEG/JPEG encoder and decoder;

Figure 2 is a block diagram illustrating prior art watermarking systems for JPEG and/or MPEG image data;

25 Figure 3 is a block diagram illustrating components of a watermarking system in accordance with an embodiment of the present invention;

Figure 4 is a block diagram of a first preferred embodiment of a QDCT watermarking system;

Figure 5 is a block diagram of a human visual system (HVS) analysis module of the watermarking system;

Figure 6 is a flow diagram of a block classification process executed by the HVS analysis module;

5 Figure 7 is a flow diagram of a mask generation process executed by the HVS analysis module;

Figure 8 is a schematic diagram of an 8x8 binary mask generated by the mask generation process;

10 Figure 9 is a block diagram of a watermark embedding module of the QDCT watermarking system;

Figures 10 to 13 are schematic diagrams illustrating a bit rate control process executed by a bit rate control module of the QDCT watermarking system;

Figure 14 is a block diagram of a preferred embodiment of a QDCT domain watermark detector;

15 Figure 15 is a block diagram of a watermark detection module of the QDCT domain watermark detector;

Figure 16 is a block diagram of a preferred embodiment of a DCT domain watermark detector;

20 Figure 17 is a block diagram of a preferred embodiment of a spatial domain watermark detector;

Figure 18 is a block diagram of a preferred embodiment of a spatial watermark embedding system for embedding a spatial watermark in the QDCT domain;

Figure 19 is a block diagram of a watermark embedding with DC-coefficient drift compensation module for the spatial watermark embedding system;

25 Figure 20 is a block diagram of a preferred embodiment of a generic watermark embedding system for embedding generic-domain watermark in the QDCT domain;

Figure 21 is a block diagram of a preferred embodiment of a DCT watermark embedding system for embedding a DCT watermark in the QDCT domain;

30 Figure 22 is a block diagram of a module for embedding generic-domain watermarks into inter-coded blocks of MPEG host data in the QDCT domain;

Figure 23 is a block diagram of a preferred embodiment of an enhanced QDCT watermark embedding system with drift compensation;

Figure 24 is a block diagram of a preferred embodiment of a QDCT blind watermark detector with drift-compensation data de-spreading;

5 Figure 25 is a flow diagram of a de-spreading matrix extraction process executed by the QDCT watermark detector; and

Figure 26 is a schematic diagram of a de-spreading matrix assembly step of the de-spreading matrix extraction process.

10 **DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS**

As shown in Figure 3, preferred embodiments of the present invention perform watermarking in the quantized DCT (QDCT) domain using a QDCT watermarking module 400. This is efficient for embedding watermark data in compressed image (*e.g.*, MPEG or JPEG) data, because it does not require computationally expensive quantization, de-quantization, DCT or inverse DCT operations to be performed. In contrast, VLC and inverse VLC operations are typically implemented by look-up tables, and are therefore fast to perform. Moreover, because the embedding is performed in the QDCT domain, the watermark data is not quantized, and is therefore more robust. Additionally, the embodiments avoid the shortcomings and limitations of embedding directly at the bit-
15 stream domain, as described above.
20

The preferred embodiments are described below in relation to MPEG and JPEG image data, with emphasis on MPEG video image data. However, it will be apparent that the invention can be applied to compressed image data in a variety of formats, including
25 H.261, H.263, MPEG 4, or any other format based on a quantized linear transform, such as a DCT transform.

A watermarking system, as shown in Figure 4, includes a VLC module 16, an inverse VLC module 24, a watermark embedding module 402, a human visual system (HVS) analysis

module 403, and a bit rate control module 405. The watermarking system embeds watermark data in DCT-compressed video (*e.g.*, MPEG, H.261, H.263) or still (*e.g.*, JPEG) image data. The inverse VLC module 24 is a standard MPEG module that extracts VLC codewords from intra-coded blocks and prediction-error (inter-coded) blocks of Y (luminance) data from a multimedia bitstream 20, and performs an inverse VLC operation to obtain 8x8 matrices of quantized-DCT coefficients, referred to as host data v . In the case of MPEG video, the host data v can be extracted from I-, B-, or P- frames. The host data v is independently processed by the watermark embedding module 402 and the HVS analysis module 403. The HVS analysis module 403 analyses the host data v to decide which parts of the data should not be watermarked in order to reduce image degradation. The watermark embedding module 402 then embeds watermark data into remaining parts of the host data v . The bit rate control module 405 compares the watermarked bitstream from the VLC module 16 with the unmarked bitstream 20 and replaces selected codewords of the watermarked bitstream with their corresponding unmarked codewords in order to reduce variations in the bit rate of the watermarked bitstream 38 from the unmarked bitstream 20.

In the described embodiments, the human visual system (HVS) analysis module 403, the QDCT-domain watermark embedding module 402, and the bit rate control module 405 of the watermarking system are implemented as software modules executed by a standard processor, such as an Intel™-based personal computer system running a Windows™ operating system, a general purpose digital signal processor (DSP) such as a Texas Instruments™ C6x DSP or an Analog Devices™ SHARC™ DSP, or an application-specific DSP such as an Oxford Micro Devices™ A236 Video DSP. The variable length coding, quantization, discrete cosine transform, and the corresponding inverse operation modules of the watermarking system are standard image processing modules, which are also implemented in software. However, it will be apparent that at least parts of the modules of the watermarking system can be implemented as dedicated hardware circuits such as programmable logic devices (PLDs) or application-specific integrated circuits (ASICs). The modules of other embodiments described below are similarly implemented.

The HVS analysis module 403 generates an 8x8 HVS-based mask G_2 for each input 8x8 QDCT block that determines which coefficients in that block can be embedded with watermark data. As shown in Figure 5, the HVS analysis module 403 includes a zero masking module 601, a block classification module 602, an edge referral module 603, an intra-coded block detector 604, a switch 605, and a multiplier 606. 8x8 QDCT blocks of host data v entering the HVS analysis module 403 are analysed by the intra-coded block detector 604. If the detector 604 determines that a block is an intra-coded block, then the block is forwarded to the block classification module 602. The block classification module 602 analyses the intra-coded 8x8 QDCT block to determine whether the block includes directed edges, generating a block mask G_B as output. The block mask G_B is used to avoid embedding watermark data in blocks containing directed edges. This reduces the perceived visual impact of the watermark on image data. The process executed by the block classification module 602 for detecting directed edges is shown in Figure 6. At step 701, partial inverse quantization is performed on the block by inverse quantizing the first row and column of the quantized DCT coefficients and the upper leftmost diagonal quantized AC DCT coefficient (*i.e.*, the 4th zig-zag-scanned quantized AC DCT coefficient) to generate the corresponding unquantized AC DCT coefficients. At step 702, the absolute values of the 1st, 2nd, and 4th zig-zag-scanned unquantized AC DCT coefficients are compared with a predetermined threshold value Th_A . If any one of these coefficients is greater than or equal to Th_A , then the host data v is deemed to contain a directed edge, and the block mask G_B is set to zero at step 705. Otherwise, the absolute values of the unquantized AC DCT coefficients in the first row and the first column are independently summed. If either sum is greater than or equal to a second predetermined threshold value Th_B , then the host data v is deemed to contain a directed edge and the block mask G_B is set to zero at step 705. Otherwise, the host data is deemed to not contain a directed edge, and the block mask G_B is set to 1. The block mask G_B is stored for subsequent use by the edge referral module 603, as described below.

Alternatively, directed edges can be detected by preprocessing the host video/image using other edge detection methods such as those described by J.F. Canny in "A computational approach to edge detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679-698, 1986, or by B.T. Hannigan, A. Reed, B. Bradley, in "Digital watermarking using improved human visual system", *Proc. SPIE Security and Watermarking of Multimedia Contents III*, vol. 4314, pp. 468-474, 2001 ("Hannigan"). This preprocessing is performed only once for each host video or image.

In the case of predicted frames of MPEG video data, a prediction-error block of a B- or P-frame is deemed to contain a directed edge in the spatial domain if its associated prediction-reference block has been deemed to contain a directed edge using the method described above. This is efficient because it avoids any need to perform edge detection on predicted frames. This edge referral process is achieved by modules 603 and 604 in Figure 5. If the intra-coded block detector 604 determines that an incoming block is not intra-coded, the edge referral module 603 identifies all the prediction-reference blocks associated with the incoming block. If the 8x8 boundaries of the incoming block are aligned with the 8x8 boundaries of its prediction-reference frame, the edge referral module 603 outputs the G_B of the associated prediction-reference block. Otherwise, if the 8x8 boundaries of the incoming block and its prediction-reference frame are not aligned, then the incoming block is associated with up to four adjoining 8x8 blocks in the prediction-reference frame. In this case, the edge referral module 603 outputs $G_B=1$ only if all the associated prediction-reference blocks have $G_B=1$, otherwise $G_B=0$ is output (*i.e.*, the incoming block is deemed not to contain an edge only if none of its associated adjoining prediction-reference blocks contains an edge).

25

The edge detection process described above can sometimes erroneously classify texture in a block as a directed edge, thus reducing the capacity available for watermark data. For applications that require a large watermark capacity, the edge detection process of Figure 6 can be modified by first performing texture detection by a known texture detection method, such as that described in Hannigan. If texture is detected in a block, then the

30

block is deemed to not contain directed edges, and the block mask G_B is set to 1. Otherwise the steps in Figure 6 are executed, as described above.

Returning to Figure 5, an 8x8 binary mask is generated by the zero masking module 601, using a mask generation process, as shown in Figure 7. The mask generation process is controlled by two user-specified parameters $FREQ_LO$ and $FREQ_HI$ that specify which parts of each 8x8 QDCT/DCT block can be watermarked. These parameters are stored in memory associated with the watermarking system. The mask generation process can be represented as a simple loop, incrementing a loop index i from a value of $i = 0$, corresponding to the DC coefficient at the top left corner of the 8x8 block to a value of 63, corresponding to the bottom right coefficient of the 8x8 block. The loop begins by setting the loop index i , representing a location within the 8x8 mask matrix, to zero at step 800. At step 801, the index is tested to determine whether it corresponds to the DC coefficient, *i.e.*, $i = 0$. If so, then the corresponding coefficient of the mask is set to zero at step 807. The loop index is then incremented at step 808. At step 809, the loop index is tested to determine whether it has reached a value of 64. If not, then the process returns to step 801. At step 802, the loop index is tested to determine whether it corresponds to a top row of the block. If so, then the corresponding coefficient of the mask is set to zero at step 807. Otherwise, the loop index is tested to determine whether it corresponds to the left column of the block at step 803. If so, then the coefficient is set to zero as described above. Otherwise, the index is tested to determine whether it is less than or equal to a value $FREQ_LO$, a user supplied parameter. If so, then the mask is set to zero. Otherwise, the loop index is tested at step 805 to determine whether it is greater than or equal to a value $FREQ_HI$, in which case its coefficient is set to zero at step 807. Otherwise, the process proceeds to step 806, where the corresponding coefficient is set to a value of 1. After all 64 mask coefficients have been processed, the test at step 809 will succeed, and the process ends.

The result of the mask generation process is an 8x8 binary mask, as shown in Figure 8. The AC coefficients in the first row 902 and first column 904 of the mask are always set to zero to avoid watermarking horizontal and vertical edges. This is an efficient and effective

method of preventing horizontal and vertical ringing distortions that otherwise occur, particularly around directed edges in the picture/frame. In order to be invisible, it is preferable that the watermark data does not contain horizontal or vertical edges in the spatial domain. This is also relevant to blind watermark detection, as a watermark detector
5 can perform edge detection on the watermarked data in order to skip “edge blocks” that do not contain watermarks. However, if the watermark itself introduces new edges, such a watermark detector will be confused. Within the remaining 7x7 square region, the coefficients within a top left triangular region 906 and a bottom right triangular region 908 are set to zero, with the remaining coefficients set to 1. The triangular regions 906, 908
10 are defined by the user-specified parameters $FREQ_LO$ and $FREQ_HI$ that define extremities of zigzag scanned AC locations. For example, the binary mask of Figure 8 corresponds to $FREQ_LO$ and $FREQ_HI$ values of 13 and 54, respectively.

Returning to Figure 5, the 8x8 binary mask generated by the zero masking module 601 is
15 multiplied in the multiplier 606 by the block mask G_B generated by the block classification module 602 to produce a mask G_2 that determines which QDCT coefficients of each 8x8 block can be watermarked. As shown in Figure 4, this HVS-based mask G_2 is output to the watermark embedding module 402 to control the watermark embedding locations.

20 The watermark embedding module 402 executes a watermark embedding process that embeds watermark data in host data v , comprising 8x8 matrices of QDCT coefficients. As shown in Figure 9, the watermark embedding module 402 includes a duplication module 501, a randomisation module 502, an absolute value module 503, and a clipping module 504. In addition to the host data v and the HVS-based mask G_2 , the watermark embedding
25 process uses a number of embedding parameters, including a spreading code c , a spreading factor N , the watermark data itself, and a user-specified gain factor G_I .

The watermark is a sequence of data values each of which is either +1 or -1, generated from binary watermark data by outputting a +1 value for each watermark data bit value of
30 1 and a -1 value for each bit value of 0. The duplication module 501 duplicates every

watermark value of +1 or -1 by the spreading factor N , which is the length of the spreading code c . The resulting data is multiplied by the spreading code c , also a sequence of +1 or -1 values, in a code multiplier module 507 to generate a spread spectrum watermark w , which is therefore also a sequence of +1 and -1 values.

5

The user specified gain factor G_1 is a positive real number that serves as a proprietary parameter that allows a watermark originator to control the quality of the embedding and detection processes. Preferably, different values of G_1 are used for intra-coded and prediction-error blocks due to their differing sensitivities to watermark distortion. In any case, variations in G_1 can be chosen to meet the needs of the watermark originator. The absolute value module 503 processes the host data v and outputs its absolute value $|v|$. This is multiplied by the gain factor G_1 and the mask G_2 in a gain multiplier 505 to determine a maximum permissible gain factor $G = G_1 G_2 |v|$, whereby scalar multiplication is performed for individual DCT coefficients of v and the corresponding entry in the G_2 matrix. The randomisation module 502 generates a gain factor matrix $rand(G)$ comprising random integers between 1 and the corresponding value of G , inclusive. However, if a value of G is zero, then the corresponding gain factor value is also zero.

This gain factor is multiplied by the spread watermark w in a watermark gain multiplier module 506 to generate embedding data $rand(G) \cdot w$. The multiplication is performed as follows. First, the spread watermark w is arranged in an 8x8 matrix format following the zig-zag scanning order. The resulting matrix is then multiplied by the $rand(G)$ matrix using scalar multiplication of corresponding entries in the two matrices. The resulting embedding data is then linearly added to the host data v in an embedding addition module 508 to generate marked data v' , subject to the additional constraints that each element in v' has the same sign as its corresponding element in v , and is clipped to ± 1 and ± 255 . The marked data v' is then variable length entropy coded by the VLC module 16, as shown in Figure 4, to produce a watermarked bit-stream.

The bit rate control module 405 processes the watermarked bit-stream from the VLC module 16 in order to control variations from the bit rate of the original un-watermarked bitstream 20. As shown in Figure 10, a buffered bitstream 1901 of input data comprises a sequence of VLC codewords 1902. After watermark embedding, the bitstream 1903 of watermarked data from the VLC module 16 comprises modified VLC codewords 1904, as shown in Figure 11. Some of the watermarked codewords 1904 are longer than their unmarked counterparts 1902 as a result of the embedded watermark data, while others may have the same or shorter bit length. For example, VLC codewords VLC1, VLC3, VLC4, and VLC5 in Figure 11 have increased in bit length, as indicated by shaded regions 1905.

10 If the total bit count of the embedded bitstream 1903 exceeds the bit count of the buffered input bitstream 1901 by more than a predetermined threshold amount, then the watermarked VLC codewords 1904 are modified as follows. First, the watermarked VLC codewords 1904 are sorted by the increased bit count of portions 1905 in reverse order, as shown in Figure 12. The VLC codeword with the longest increased portion 1905 is replaced with its original un-watermarked VLC codeword. For example, the fourth watermarked codeword 1908 is replaced with the corresponding codeword 1906 from the unwatermarked bitstream 1901, as shown in Figure 13. The total quantity (*i.e.*, number of bits) of data of the resulting bit-stream 1910 is then redetermined. If this still exceeds the quantity of data in the original bit-stream 1901 by more than the threshold amount, then

20 the next VLC codeword 1912 in the ordered list is restored. This process is continued until the resulting bitstream 1910 falls within the required limit. The size of the buffer 1901 can be any size from a macroblock to the entire image or video frame, depending on the user applications or requirements. Compared to the bit rate conservation scheme described in F. Hartung and B. Girod, *Watermarking Of Uncompressed And Compressed Video*, in

25 *Signal Processing*, vol. 66, pp 283-301, May 1998 and US Patent 5,809,139 ("Hartung") which checks and restores VLC codewords individually, the bit rate control method described herein provides more flexibility in reducing the resultant loss in data hiding capacity, as well as improving the strength of the surviving watermark.

Once a watermarked bitstream 38 has been generated, the hidden watermark data can be extracted by a watermark detector employing either standard non-blind detection based on subtraction of unwatermarked host data, or blind detection based on correlation and pre-filtering, as described in “Hartung” and U.S. Patent 5,809,139. Watermark detection can
5 be performed in any domain. For watermark detection in the quantized DCT-domain, a watermark detector, as shown in Figure 14, is used that includes the inverse VLC module 24 and a watermark detection module 1002. The watermark detector optionally includes the HVS analysis module 403 to generate an HVS-based mask G_2 that indicates where the watermark data is embedded. This requires knowledge of the user-specified parameters
10 $FREQ_LO$, $FREQ_HI$, Th_A and Th_B that were used to embed the watermark data, as described above. The watermark detection module 1002 despreads the watermarked data using a synchronised copy of the spreading code c .

As shown in Figure 15, the watermark detection module 1002 includes a correlation
15 module 1303, and a thresholding module 1305. The correlation module 1303 includes a summation module and a dump module 1306. If the HVS-based mask G_2 is available, it is used to multiply the spreading code c in order to set the elements in c to zero at locations where no watermark data was embedded during the watermark embedding process. The correlation module 1303 then multiplies the resultant spreading code by the marked host
20 data v' and the summation module 1304 of the correlation module 1303 sums the product to generate a correlation sum. If G_2 is not available, the original spreading code c is multiplied by the marked host data v' to obtain the correlation sum. After this correlation sum has accumulated N product terms, the dump module 1306 sends its value to the thresholding module 1305, after which the correlation sum is reset to zero for detecting the
25 next watermark data bit. The thresholding module 1305 performs thresholding on the correlation sum to generate a value for the watermark data, or alternatively an indicator of its presence. For example, if the correlation sum is positive or larger than a user-specified positive value, the thresholding module outputs a watermark data bit of +1, and otherwise outputs a watermark data bit of 0. Alternatively, for detecting the presence of unsigned
30 watermark data, the thresholding module outputs a watermark presence indicator if the

correlation sum exceeds a second user-specified positive-valued threshold value. Alternatively, other thresholding and detection rules can be designed based on the specified needs of the watermark originator, and the nature of the embedded watermark data.

5

To detect watermark data in the DCT domain, a DCT watermark detector, as shown in Figure 16, is used. The DCT watermark detector includes the inverse VLC module 24, a first inverse quantization module 26, a correlation module 1603 and a thresholding module 1605. The DCT watermark detector optionally includes the HVS analysis module 403 and
10 a second inverse quantization module 1106, which is functionally identical to the first inverse quantization module 26. An input watermarked bit-stream 38 is inverse variable length coded by the inverse VLC module 24 and unquantized by the first inverse quantization module 26. The quantized DCT data from the inverse VLC module 24 is optionally processed by the HVS analysis module 403 to generate the HVS-based mask G_2
15 indicating the quantized DCT locations that do not contain watermark data. This is then multiplied by the spreading code c . The resultant spreading code can be inverse quantized, if desired, by the second inverse quantization module 1106 and is provided with the DCT domain watermarked data and the spreading factor N to the correlation module 1603. The correlated output is sent to the thresholding module 1605 which performs thresholding to
20 reconstruct the watermark data, as described above.

To detect watermark data in the spatial domain, a spatial domain watermark detector, as shown in Figure 17, additionally includes first 28 and optionally second 1208 inverse DCT modules. The spreading code c is inverse quantized by the second inverse quantization
25 module 1106 and then inverse DCT transformed by the second 1208 inverse DCT module to generate $Q^{-1}(DCT^{-1}(c))$. The sign (polarity) of $DCT^{-1}(Q^{-1}(c))$ is then determined by a sign module 1307 for despreading of the spatial domain watermarked data by a correlation module 1703 and a thresholding module 1705 in a similar manner to that described above.

Alternative embodiments can be used to embed, in the quantized DCT domain, additive watermarks that were originally intended to be added in other domains. The basic principle is to transform the other-domain watermark to the QDCT domain, generating what is referred to as the equivalent QDCT watermark, before embedding the latter into the host data. However, as the equivalent QDCT watermark can contain both DC and AC coefficients, the process of embedding these coefficients into intra-coded blocks of the host data is handled separately. This is because the DC and AC QDCT coefficients of intra-coded blocks have different digital encoding formats: DC coefficients are coded by differential pulse code modulation (DPCM) followed by decimal-to-binary conversion, whereas AC coefficients are encoded by VLC, as described above.

For example, a spatial watermark embedding system, as shown in Figure 18, includes the DCT module 12, the quantization module 14, a de-multiplexer 1402, and a watermark embedding with DC-coefficient drift compensation module 1404. A spatial domain watermark is transformed to the DCT frequency domain by the DCT module 12 and then quantized by the quantization module 14 to generate equivalent watermark data w_{equiv} in the QDCT domain, which is divided into AC and DC coefficients by the de-multiplexer 1402 for separate embedding by the watermark embedding with DC-coefficient drift compensation module 1404.

20

As shown in Figure 19, the watermark embedding with DC-coefficient drift compensation module 1404 includes a de-multiplexer 1420, the inverse VLC module 24, the VLC module 16, a multiplexer 1418, the clipping module 504, a decoding module 1408, an encoding module 1416, a delay module 1412, and a switch 1414. The de-multiplexer 1420 separates the input MPEG/JPEG bitstream into digital AC and DC components, referred to as the input AC bitstream and DC bitstream, respectively. The inverse VLC module 24 performs inverse entropy encoding on the input AC bitstream to generate AC quantized DCT host data v . An addition module 1406 adds this to the AC coefficient of w_{equiv} to generate watermarked AC data v' , which is then entropy encoded by the VLC module 16 to generate a watermarked AC bitstream. The input DC bitstream is converted by a

30

decoding module 1408 into quantized analog values. A summing module 1410 adds these to the DC coefficient of w_{equiv} to generate watermarked DC data, and subtracts the DC coefficient of w_{equiv} of the *previous* 8×8 block *if* the current block is not the starting block of a slice. This conditional subtraction operation is implemented using the delay module 5
1412 and the switch module 1414. Its purpose is to achieve DC drift compensation by preventing the visual distortions introduced by the DC of the watermark from accumulating onto subsequent 8×8 blocks as a result of the inverse DPCM operation performed during video playback. The resultant DC-drift-compensated watermarked DC data is encoded back into a digital bitstream by an encoding module 1416, and multiplexed
10 with the watermarked AC-component bitstream by the multiplexer module 1418. The clipping module 504 clips the digital bitstream within ± 1 to ± 255 to produce the final bitstream 38 with spatial watermark.

Additive watermarks in other linear transform domains, such as the Fourier domain, can
15 also be embedded by including a watermark translation module 1500, as shown in Figure 20. The watermark translation module 1500 translates the watermark into the spatial domain. The spatial domain watermark is then embedded as described above.

Similarly, a DCT watermark embedding system, as shown in Figure 21, is the same as the
20 spatial watermark embedding system of Figure 18, but without the DCT module 12. A DCT watermark is quantized by the quantization module 14 to generate the QDCT equivalent watermark data w_{equiv} which is then embedded, as described above.

Embedding of other-domain watermarks (*e.g.*, spatial, DCT, Fourier and other linear-
25 transform-domain watermarks) in inter-coded blocks of MPEG bit-streams in the quantized DCT domain is relatively more straightforward because all 8×8 QDCT coefficients of an inter-coded block, whether DC or AC QDCT coefficients, are encoded together by variable length entropy encoding. There is no separate encoding for the DC QDCT coefficients. For this case, as shown in Figure 22, the inverse VLC module 24
30 performs inverse entropy encoding on input inter-coded blocks to generate 8×8 quantized

DCT host data v . This is linearly added to the 8×8 coefficients of w_{equiv} (obtained as in Figures 14, 15 and 16) to generate watermarked data v' , which is then entropy encoded by the VLC module 16 and clipped to ± 1 and ± 255 by the clipping module 504 to generate a watermarked bit-stream for MPEG inter-coded blocks.

5

In yet a further embodiment, the watermarking system consists of a QDCT watermark embedding module with drift compensation, and a QDCT blind watermark detector enhanced with drift-compensation data de-spreading. This embodiment provides a drift-compensated watermarking system with enhanced blind detection performance.

- 10 As shown in Figure 23, the enhanced QDCT watermark embedding module with drift compensation includes the components of the QDCT watermarking module 400, a drift-compensation module 2102 as proposed in "Hartung", an inverse VLC module for both intra- and inter-coded blocks 2104, and the bit rate control module 405. The drift-compensation module 2102 executes a drift compensation process, as described in
- 15 "Hartung". Briefly, for the watermarked QDCT data v' in every prediction-reference macroblock that has been embedded with watermark data by the QDCT watermark embedding module 402, the drift-compensation module 2102 computes the spatial equivalent of $-(v'-v)$ where v is the associated unwatermarked data, and embeds it into all prediction-error macroblocks associated with the prediction-reference macroblock, using
- 20 information obtained from the motion vectors of the host data. Since $(v'-v)$ constitutes the watermark data embedded into the prediction-reference frames, the drift-compensation process effectively embeds the same spread watermark, but with negative polarity, into the predicted frames, so as to cancel the visual distortions induced by $(v'-v)$ which would otherwise propagate or "drift" into the predicted frames due to motion predictions
- 25 performed during video playback. If a predicted macroblock does not originally contain prediction-error data (commonly referred to as "skipped" macroblocks in MPEG nomenclature), then the drift compensation process creates the prediction-error data containing the negative-polarity watermark as described above and changes the skipped macroblock into a prediction-error macroblock.

The negative-polarity watermark embedded in the drift-compensated predicted frames provides additional information about the same watermark data that is hidden in the associated prediction-reference frames. This information can be used by the QDCT watermark detector with drift-compensation data de-spreading, as shown in Figure 24, to
5 improve the overall reliability of blind watermark detection.

The enhanced watermark detector includes the inverse VLC for prediction-error blocks module 1701, the inverse VLC for motion vectors module 1702, a de-spreading sequence submatrix extraction module 1803, a correlation module 1805, an accumulator 2206, a thresholding module 1806, and the components of the watermark detection module 1002
10 shown in Figure 15, excluding the thresholding module 1305. The correlation module 1805 includes a multiplier module 2202 and a summation module 2204.

The inverse VLC for prediction-error blocks module 1701 performs inverse VLC coding of drift-compensated predicted frame data to generate drift-compensated prediction-error QDCT blocks. Similarly, the inverse VLC coding motion vector module 1702 generates
15 motion vectors from the predicted frame data. For every watermark bit to be de-spread from a prediction-reference frame, the associated motion vectors are used to identify all the associated drift-compensated prediction-error blocks. For each drift-compensated prediction-error block in the predicted frame, the motion vectors are used to identify the associated prediction-reference block from the prediction-reference frame. A de-spreading
20 matrix corresponding to each prediction-reference block is then determined from the main despreading sequence c and the motion vectors by a de-spreading matrix extraction module 1803.

The de-spreading matrix extraction module 1803 executes a de-spreading matrix extraction process, as shown in Figure 25. The main de-spreading sequence c associated with the
25 prediction-reference block of concern is first multiplied at step 2602 by the HVS-based mask G_2 , obtained as shown earlier in Figure 14, to set the spreading sequence elements associated with non-embedding QDCT coefficients to zero. This results in a *possible* de-spreading matrix. At step 2604, the motion vector is used to determine whether the prediction-error block boundary is aligned with the 8x8 block boundary of the associated

prediction-reference frame. If so, the de-spreading matrix obtained above is sent directly to the output. Otherwise, the required de-spreading matrix C is assembled at step 2606 from the de-spreading matrices of up to four neighbouring blocks adjacent to the current prediction-reference block of concern. For example, as shown in Figure 26, an 8×8 predicted block E is not aligned with its prediction-reference frame, and it overlaps with 8×8 blocks B_1 , B_2 , B_3 and B_4 in the prediction-reference frame. In order to de-spread the negative-polarity watermark embedded in E , the necessary de-spreading matrix C is assembled from the de-spreading matrix elements associated with the four different corners of B_1 to B_4 , shown as different shaded regions in Figure 26. To obtain C from the de-spreading matrices of blocks B_1 to B_4 , the following steps are performed:

- (i) Starting with block B_1 , the 8×8 de-spreading matrix of block B_1 is pre-multiplied by the 8×8 DCT transform of a matrix P_1 and post-multiplied by the 8×8 DCT transform of another matrix Q_1 to generate a 8×8 matrix C_1 .
- (ii) The above process is repeated for blocks B_2 to B_4 , using P_2 to P_4 and Q_2 to Q_4 respectively, so as to obtain another three 8×8 matrices C_2 to C_4 respectively.
- (iii) Matrices C_1 to C_4 are then linearly summed together, followed by sign (polarity) extraction to obtain the de-spreading matrix C for the predicted block E .
- (iv) The matrices P_1 to P_4 and Q_1 to Q_4 are:

$$\begin{aligned}
 P_1 &= \begin{bmatrix} 0 & I_a \\ 0 & 0 \end{bmatrix} & Q_1 &= \begin{bmatrix} 0 & 0 \\ I_b & 0 \end{bmatrix} \\
 P_2 &= \begin{bmatrix} 0 & I_a \\ 0 & 0 \end{bmatrix} & Q_2 &= \begin{bmatrix} 0 & I_{8-b} \\ 0 & 0 \end{bmatrix} \\
 P_3 &= \begin{bmatrix} 0 & 0 \\ I_{8-a} & 0 \end{bmatrix} & Q_3 &= \begin{bmatrix} 0 & 0 \\ I_b & 0 \end{bmatrix} \\
 P_4 &= \begin{bmatrix} 0 & 0 \\ I_{8-a} & 0 \end{bmatrix} & Q_4 &= \begin{bmatrix} 0 & I_{8-b} \\ 0 & 0 \end{bmatrix}
 \end{aligned}$$

where I_n is a $n \times n$ identity matrix, and a and b are the pixel-height and pixel-width of the overlapped bottom-right corner of prediction-reference block B_1 respectively, as shown in Figure 27.

When viewed in the spatial domain, the pre-multiplication with $DCT(P_i)$ and post-multiplication with $DCT(Q_i)$ have the effects of extracting the spatial-domain de-spreading sub-matrix for the corresponding corner of block B_i and translating it to a different corner
5 in the final required spatial-domain de-spreading matrix for the predicted block E , as shown in Figure 26. The DCT of P_1 to P_4 and Q_1 to Q_4 can be pre-computed for all possible combinations of a and b , and stored in the memory of the watermark detector for computational efficiency.

Returning to Figure 24, the de-spreading matrix C is used to de-spread the prediction-error
10 blocks, which contain the negative polarity watermark associated with the de-spreading matrix C . The resulting output from the correlation module 1805 is referred to as the drift-compensation correlation sum Y_{dc} . This process is carried out for *all* prediction-error blocks associated with a watermark bit undergoing correlation detection, including the bi-directional prediction-error blocks and half-pixel prediction-error blocks. All the resultant
15 values of Y_{dc} are accumulated by an accumulator module 2206 and then subtracted from the prediction-reference correlation sum Y_{pr} (the output of the correlation module 1303 shown in Figure 15) retrieved from the associated prediction-reference frame(s) to generate an enhanced correlation sum Y . The watermark data is finally reconstructed by thresholding in the thresholding module 1806.

20

Many modifications will be apparent to those skilled in the art without departing from the scope of the present invention as herein described with reference to the accompanying drawings.

CLAIMS:

1. A method for use in watermarking image data, including generating watermarked quantized image data by linearly adding watermark data to quantized image data.
5
2. A method as claimed in claim 1, wherein said method includes clipping the absolute value of said watermarked quantized image data to be greater than or equal to 1 and less than or equal to 255, and changing the sign of said watermarked quantized image data if said sign is different to the sign of said quantized image data.
10
3. A method as claimed in claim 1 or 2, wherein the method includes generating spread spectrum watermark data from input watermark data using a spread spectrum technique.
- 15 4. A method as claimed in claim 3, wherein the step of generating spread spectrum watermark data includes generating intermediate watermark data by duplicating each bit of the input watermark data by a predetermined spreading factor, and multiplying the intermediate watermark data by a predetermined spreading code.
- 20 5. A method as claimed in claim 3, wherein the step of generating spread spectrum watermark data includes generating intermediate watermark data by duplicating each bit of the input watermark data by a predetermined spreading factor, and multiplying the intermediate watermark data by a predetermined spreading code and a gain factor.
- 25 6. A method as claimed in claim 5, wherein the gain factor is zero if a maximum gain value is zero, and otherwise the gain factor is a random integer between 1 and the maximum gain value, inclusive.
- 30 7. A method as claimed in claim 6, wherein the maximum gain value is the product of an adjustable gain parameter, a mask generated on the basis of a human visual system

analysis of the quantized image data, and the absolute value of the quantized image data.

8. A method as claimed in claim 7, wherein the mask is also generated on the basis of at
5 least one adjustable mask parameter.
9. A method as claimed in claim 1, wherein the quantized image data includes quantized transform coefficients.
- 10 10. A method as claimed in claim 1, wherein the quantized image data includes quantized discrete cosine transform (DCT) coefficients.
11. A method as claimed in claim 1, including generating said watermark data in a quantised DCT domain from second watermark data in a different linear transform
15 domain.
12. A method as claimed in claim 11, wherein said step of generating said watermark data includes adding AC coefficients of said watermark data to respective AC coefficients of said image data, and adding a DC coefficient of said watermark data to DC
20 coefficients of said image data and subtracting the DC coefficient of said watermark data added to the previous block of said image data if the current block is not the starting block of a slice of said image data.
13. A method as claimed in claim 11 or 12, wherein said different linear transform domain
25 includes one of spatial, DCT, and Fourier domains.
14. A method as claimed in claim 9, including selecting quantized transform coefficients for watermarking, and said step of generating includes generating watermarked quantized transform coefficients by linearly adding watermark data only to the selected
30 quantized transform coefficients.

15. A method as claimed in claim 14, wherein said step of selecting includes selecting quantized transform coefficients on the basis of their respective frequencies.
- 5 16. A method as claimed in claim 15, wherein said step of selecting includes not selecting quantized transform coefficients corresponding to DC frequency.
17. A method as claimed in claim 14, wherein said step of selecting includes selecting quantized transform coefficients on the basis of their position within a transform
10 matrix.
18. A method as claimed in claim 17, wherein said step of selecting includes not selecting coefficients in the top row and the left column of a matrix of quantized DCT coefficients.
- 15 19. A method as claimed in claim 14, wherein said step of selecting includes selecting quantized transform coefficients on the basis of at least one adjustable parameter.
20. A method as claimed in claim 19, wherein said step of selecting includes selecting
20 quantized transform coefficients whose zig-zag scan indices are in a range defined by first and second adjustable parameters.
21. A method as claimed in claim 14, wherein said step of selecting includes selecting quantized transform coefficients on the basis of directed edge detection.
- 25 22. A method as claimed in claim 14, wherein said step of selecting includes not selecting a block of quantized transform coefficients if the block contains a directed edge.
23. A method as claimed in claim 14, including performing texture detection on a block of
30 transform coefficients and selecting the block if the block contains texture, and

otherwise performing directed edge detection on the block and selecting the block if it does not contain a directed edge.

24. A method as claimed in claim 22, including deeming that a predicted block contains a
5 directed edge if any of the up to four corresponding reference blocks contains a directed edge.
25. A method as claimed in claim 21 or 22, including deeming that a block of discrete
cosine transform (DCT) coefficients contains a directed edge if the absolute value of
10 any of the 1st, 2nd and 4th zig-zag scanned unquantized AC DCT coefficients in the block is greater than or equal to a first predetermined value, or if the sum of the absolute unquantized values of the top row or the left column of the block is greater than or equal to a second predetermined value.
- 15 26. A method as claimed in claim 1 or 2, wherein said method includes generating said quantized image data by inverse entropy coding input codewords.
27. A method as claimed in claim 26, wherein said method includes entropy coding said
20 watermarked quantized image data to generate output codewords.
28. A method as claimed in claim 27, wherein said method includes replacing an output
codeword with the corresponding input codeword if the increase in the total length of
output codewords from the total length of the corresponding input codewords exceeds
a predetermined value.
- 25 29. A method as claimed in claim 28, wherein said method includes replacing output
codewords with corresponding input codewords until the increase in the total length of
output codewords from the total length of the corresponding input codewords is less
than a predetermined value.

30. A method as claimed in claim 28, wherein said method includes generating a list of output codewords sorted on the basis of the increase in the length of each output codeword from the length of the corresponding input codeword, and replacing the output codeword with the greatest length difference with the corresponding input
5 codeword until the increase in the total length of output codewords from the total length of the corresponding input codewords is less than a predetermined value.
31. A method as claimed in claim 1, wherein said quantized image data includes at least one of JPEG, MPEG, H.261, and H.263 data.
10
32. A method as claimed in claim 1, wherein said quantized image data includes MPEG data including predicted frames and prediction reference frames, and said method includes performing drift compensation for watermarked image data in the prediction reference frames.
15
33. A method as claimed in claim 32, wherein said step of performing drift compensation includes subtracting, from quantized image data in predicted frames, watermark data added to quantized image data in the corresponding prediction reference frames.
- 20 34. A method for detecting a watermark in watermarked image data generated by linearly adding watermark data to quantized image data, including correlating the watermarked image data with a predetermined spreading code.
- 25 35. A method for detecting a watermark in watermarked image data generated by linearly adding watermark data to quantized image data, including inverse quantizing said watermarked image data, and correlating the inverse quantized watermarked image data with a predetermined spreading code.
- 30 36. A method for detecting a watermark in watermarked image data generated by linearly adding watermark data to quantized image data, including inverse quantizing said

watermarked image data and a spreading code, and correlating the inverse quantized watermarked image data with the inverse quantized spreading code.

37. A method for detecting a watermark in watermarked image data generated by linearly
5 adding watermark data to quantized image data, including inverse quantizing and
inverse transforming the watermarked image data and a spreading code, and correlating
the inverse quantized and inverse transformed watermarked image data with the sign of
an inverse quantized and inverse transformed spreading code, said spreading code
comprising a sequence of two values having opposite sign.
- 10 38. A method for detecting a watermark in watermarked image data generated by linearly
adding watermark data to prediction reference frames of quantized image data and drift
compensation data to predicted frames of said quantized image data, said method
including performing correlation on the reference frames to generate reference
15 correlator output, performing correlation on predicted watermark data corresponding to
said predicted frames to generate predicted correlator output, subtracting the predicted
correlator output from the reference correlator output, and thresholding the result.
- 20 39. A method for detecting a watermark in watermarked image data generated by linearly
adding watermark data to prediction reference frames of quantized image data, and
drift compensation data to predicted frames of said quantized image data, including
generating a despreading matrix for a predicted block of image data from despreading
matrices for up to four corresponding reference blocks of the watermarked image data.
- 25 40. A method for detecting a watermark in watermarked image data generated by linearly
adding watermark data to quantized image data, including selecting quantized
transform coefficients of said watermarked image data, and correlating the selected
coefficients with a predetermined spreading code.

41. A method as claimed in claim 40, wherein said step of selecting includes selecting quantized transform coefficients on the basis of their respective frequencies.
42. A method as claimed in claim 41, wherein said step of selecting includes not selecting
5 quantized transform coefficients corresponding to DC frequency.
43. A method as claimed in claim 40, wherein said step of selecting includes selecting quantized transform coefficients on the basis of their position within a transform matrix.
10
44. A method as claimed in claim 43, wherein said step of selecting includes not selecting coefficients in the top row and the left column of a matrix of quantized DCT coefficients.
- 15 45. A method as claimed in claim 40, wherein said step of selecting includes selecting quantized transform coefficients on the basis of at least one adjustable parameter.
46. A method as claimed in claim 45, wherein said step of selecting includes selecting quantized transform coefficients whose zig-zag scan indices are in a range defined by
20 first and second adjustable parameters.
47. A method as claimed in claim 40, wherein said step of selecting includes selecting quantized transform coefficients on the basis of directed edge detection.
- 25 48. A method as claimed in claim 40, wherein said step of selecting includes not selecting a block of quantized transform coefficients if the block contains a directed edge.
49. A method as claimed in claim 40, including performing texture detection on a block of transform coefficients and selecting the block if the block contains texture, and

otherwise performing directed edge detection on the block and selecting the block if it does not contain a directed edge.

50. A method as claimed in claim 48, including deeming that a predicted block contains a
5 directed edge if any of the up to four corresponding reference block contains a directed edge.
51. A method as claimed in claim 47 or 48, including deeming that a block of discrete
10 cosine transform (DCT) coefficients contains a directed edge if the absolute value of any of the 1st, 2nd and 4th zig-zag scanned unquantized AC DCT coefficients in the block is greater than or equal to a first predetermined value, or if the sum of the absolute unquantized values of the top row or the left column of the block is greater than or equal to a second predetermined value.
- 15 52. A system having components for executing the steps of any one of claims 1 to 51.
53. Software having program code for executing the steps of any one of claims 1 to 51.
54. A computer readable storage medium having program code stored thereon for
20 executing the steps of any one of claims 1 to 51.
55. A watermarking system, including a module for generating watermarked quantized image data by linearly adding watermark data to quantized image data.
56. A system for generating watermarked quantized image data, including a module for
25 selecting portions of image data on the basis of a human visual system analysis of the image data, and a module for embedding a watermark in the selected portions of the image data by linearly adding watermark data to quantized image data.

57. A system as claimed in claim 55 or 56, including a module for entropy coding the watermarked quantized image data.

58. A system as claimed in claim 57, including a module for generating the quantized
5 image data by inverse entropy coding input image codewords.

59. A system as claimed in claim 58, including a module for controlling the difference of the length of said watermarked quantized image data relative to the length of said input image data.

10

60. A system for detecting a watermark in watermarked quantized image data generated by linearly adding watermark data to quantized image data, said system including a detection module for detecting said watermark in a quantized domain by correlation and thresholding.

15

61. A system as claimed in claim 60, including a module for selecting portions of quantized image data on the basis of a human visual system analysis of the image data, and wherein said detection module detects the watermark in the selected portions of the quantized image data by correlation and thresholding.

20

62. Watermarked image data generated by executing the steps of one of claims 1 to 33.

63. A computer readable storage medium having stored thereon watermarked image data generated by executing the steps of one of claims 1 to 33.