INTERNATIONAL ORGANIZATION FOR STANDARDIZATION
ORGANISATION INTERNATIONAL DE NORMALISATION

ISO/TC97/SC2/WG8

CODED REPRESENTATION OF PICTURE AND AUDIO INFORMATION

ISO/TC97/SC2/WG8 N636
November 1987

SOURCE: ISO/TC97/SC2/WG8

TITLE: Liaison between ISO/TC97/SC2/WG8 and CCITT SGXV (Mr Okubo)

During the ISO/CCITT Joint Photographic Expert Group (JPEG)* meeting, in Washington 26-30 October 1987, it was felt that a formal liaison between the JPEG and the CCITT SGXV working groups was urgently required.

JPEG is currently aiming at a single coding algorithm for still images to be agreed end of January 1988, and as far as JPEG knows, SGXV is also aiming at a recommendation on a unified algorithm for videoconferencing services at 384 kbit/s, during 1988.

At our a June '87 meeting in Copenhagen the three techniques given below were selected for further development -

1. ADCT N502
2. ABAC N509
3. GBTC N510

A full descriptions of the techniques is given in the documents annexed to this liaison letter.

The ADCT technque, N502, in particular might be worth considering for compatibility purposes, as apart from interframe coding we note the following commonalities:

1. DCT 8x8
2. Zig-Zag Scanning
3. Uniform quantization
4. Thresholding
5. Huffman Code Tables

We would like you to consider those common parts to see if for the intraframe coding mode we might be able to harmonize our specifications, which it is felt that would be beneficial to both parties.

*JPEG is the Joint Photographic Experts Group set up by ISO TC97/SC2/WG8 and CCITT SGVIII NIC(JPEG).

ISO

INTERNATIONAL ORGANISATION FOR STANDARDIZATION

ORGANISATION INTERNATIONALE DE NORMALISATION

ISO/TC97/SC2/WG8

CODED REPRESENTATION OF PICTURE
AND AUDIO INFORMATION

ISO/TC97/SC2/WG8 N 690

/adctg N

Source: ISO / ADCTG

Title : Adaptive Discrete Cosine Transform Coding Scheme
for Still Image Telecommunication Services

Dear Hiroshi,
please find, herein, the documentation on ISO-ADCT.
It, very likely still remains few errors ... of different kinds. But this is the best issue we have been able to achieve by January the 8th.

Very Kind Regards

Alain Léger

copie: Henning POULSEN
KTAS

WG8 N640

ISO
ADAPTIVE DISCRETE COSINE TRANSFORM CODING SCHEME FOR STILL IMAGE TELECOMMUNICATION SERVICES

issued on january, 8th, 1988

Published by ISO/TC97/SC2/WG8 - ADCTG

75

ABSTRACT

This documentation presents the state of development of the Adaptive Discrete Cosine Transform (ADCT) coding technique, as proposed on january the 8 th, 1988, by ISO - TC97/sc2/wg8/adctg. A previous version said " the june version" or n502, was presented in Copenhagen (june 87) during an ISO - TC97/ sc2/wg8/jpeg meeting.

The main steps forward relate to :

- enhancements of the picture quality at very low bit rate (e.g. 0.08 bpp and 0.25 bpp)
- enhancements of the psychovisual impression of progressive update
- introduction of predefined Variable Length Code tables as default values of an adaptive entropy coding scheme, in order to gain full symetry of the Codec
- production of a replica of the original image at about 8.0 bit per pel in a final stage of a progressive or sequential updates

Note, also, that in any case this has not to be considered as a final version, refinements - open or not in a final recommendation - are still possible in some places, but, as expected, without modifications to the basic coding structure.

This documentation is the expression of a collaborative work within the ISO - TC97/sc2/wg8 adct group of experts from AT&T - Bell labs, CCETT, CSELT, DEC, KTAS, NEC and NTT.

Contents

1.0 General

1.1 INTRODUCTION

Numerous ADCT (Adaptive Discrete Cosine Transform) Coding schemes known so far, have always produced very good results at low bit rate and were very often considered as First Class Compression techniques. This has led to a great variety of implementations on a worldwide scale that in turn have entailed much consideration at international standardization and collaboration levels : CCITT, CCIR, CEPT, RACE, ESPRIT,...

As a matter of facts, the ADCT coding technique remains one of the leading coding schemes of the "first generation" for late 80's and very likely the 90's , for very wide applications, from still image services up to live image services (visiophony, visioconference and digital TV).

The technique we are thereafter describing is one of this COMPATIBLE family, specifically designed STILL grey or colour images. Its applicability ranges from low end Photovideotex services, up to Graphics arts, Press and Medical applications.

Special care has been taken to design a SIMPLE compression scheme that proves very EFFICIENT. This of much concern for the implementors of the technique who have to understand and to master EASILY.

Furthermore, the REAL TIME constraints of still image services for the ISDN or higher data rate channel, have also, always been considered, for both the coder and the decoder, and as a consequence the coding technique is fully SYMMETRICAL.

At the editing or coding level, the technique is fully ADAPTIVE to the image content and to the SERVICE REQUIREMENTS.

1.2 BRIEFING ON PRINCIPLES OF THE CODING ALGORITHM

TECHNICAL

The technique is simple and is basically funded on the following principles

1. NO IMAGE MODELS are asssumed, either in the spatial or the transform domain
2. PSYCHOVISUAL THRESHOLDS are used for the selection of the most relevant coefficients in the transform domain – by implicit thresholding of the lowest decision level – .
3. LINEAR QUANTIZATION with unit step equals to the PSYCHOVISUAL threshold of the related coefficient to be quantized.

4. LOSSLESS coding of the quantized coefficients and their relative positions – via predefined Variable Length Code tables.

5. ADAPTIVITY is achieved by suitable thresholding matrices for a particuliar set of components (e.g. Y, Cr, Cb) and resolution of the image to be coded, and by carefully predefined or customized Variable Length Code tables.

From these foundations, it follows that :

6. COMPRESSION is chiefly achieved by a LOSSLESS scheme, applied on very stable statistics in the transform domain

7. QUANTIZATION is performed at the nominal picture quality, for mini – mizing the VISIBLE Quantization ERROR.

SERVICE

° The PICTURE QUALITY and consequently the COMPRESSION factor is varied upon changing linearly the Psychovisual thresholding matrices – one Central point of the algorithm – , which act as adaptive zonal sampl – ing and as adaptive quantizing laws.

° FLEXIBILITY for FUTURE improvements is provided very easily and at insignificant cost, by the possible Redefinition of VLC tables and of Thresholding matrices through overhead information – e.g. according to progress in psychovisual perception and understanding of image.

° VERSATILITY for not yet defined or unforseen applications, by the same means outlined in (2)

° The SEQUENTIAL and PROGRESSIVE buid up of the received image, are available through the appropriate sequencing of the transformed and quantized coefficients in a large number of display stages as required by the service.

° For LOW data RATE channels ,e.g.1200 bit/s on PSTN, very low coding rates are available, i.e 0.08 bit/pel,....,0.25bit/pel, through a NON – MANDATORY preprocessing structure that is embedded in the general coding structure, and that caters for good recognisability of the picture at the early stages of the progressive up – date.

1.3 OVERVIEW OF THE FUNCTIONALITIES

1.3.1 Coding Structure for progressive update

1.3.1.1 Introduction

In some applications, low bit – rate coding of images is utilized not to repre – sent pictures in final form, but instead as an interim step in a progressive transmission that allows the viewer to have something to look at far earlier than if the image were sent with final resolution and quality in only one stage of coding. One typical application migth be Photovideotex on narrow – band channels (e.g. 4800 bit/s). Other applications, not constrained by band limitations (e.g. > 64 kbit/s), may desire to display a picture in a more pleasing way than suc – cessive rows of full resolution decoded image (i.e. sequential update), and so to display in a certain number of full scans, giving a better subjective impression.

In the following, these two kinds of aspects are considered and are globally called "Progressive Coding Structure". In fact, it is composed of a "hierarchical structure" that may not be mandatory and a "spectral selection structure" – as previously described in n502 – which may give at each MAIN – stage of the "hi – erarchical structure" a large number of progressive SUB – stages by sending, for example, one coefficient per block at a time.

Finally, in such cases, we may wish to code in the early stages at equivalent bit – rates very much smaller than that required for final picture quality. For ex – ample, progressively better quality color images might be sent, respectively, at 0.08 0.25 0.75 and finally 2.25 bits per pel.

At such low early – stage bit – rates, reproduced subjective image quality of unadorned transform coding is relatively poor compared with the original picture. In fact, a better approach usually is to low – pass filter and subsample the ori – ginal picture before early – stage coding, in order to reduce the number of pels that have to be coded. The transform can then be carried out at a higher bit – rate per transmitted pel, and the image degradation due to low – pass filtering will usually be far more acceptable than the block boundary distortion due to coding at a very low bit – rate per transmitted pel.

. This is similar to the approach used in classical pyramid coding . Coding always takes place at a relatively high bit – rate per transmitted pel. The number of pels sent is simply reduced to achieve the desired bit – rate per image.

Having already sent a low resolution version of the picture at one stage, it is not necessary to start from scratch in sending the higher quality picture at the next stage. The earlier stage quantized/decoded image can be up – sampled, with appropriate interpolating filters, and used as a prediction for the coding at the next stage. We simply substract the prediction from the original, and transform code the differential pel blocks as described above.

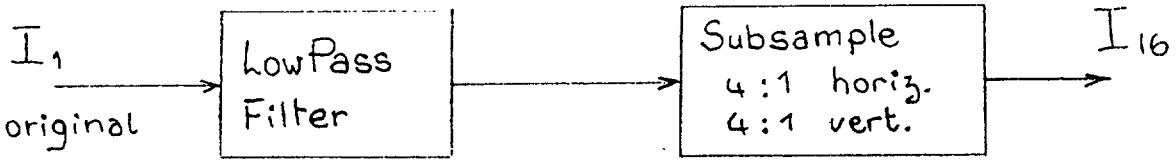
Note that this is different than traditional pyramid coding, in which an unquantized/uncoded low – pass filtered image is substracted to get the differential image to be transmitted. With this method, coding distortion at an early stage cannot be corrected at a later stage, unlike the presently described technique.

At the earlier stages, the picture quality of pyramid transform coding is no – ticeably better than with either transform coding. However, at the final stage, there may be discrepancies between this hierarchical coding structure and one – pass transform coding at the final bit – rate. Nevertheless, the full match with

"hierarchical" (i.e. progressive) coding and "one – pass" (i.e. sequential) coding of the same picture is ALWAYS possible, by simply reordering the data in a buffer.

1.3.1.2 Principles of operation of the hierarchical structure

First, the image is low – pass filtered and subsampled, for example, by 4:1 in both directions to give an overall reduction of 16:1 in the number of pels. We call this image I_{16} :

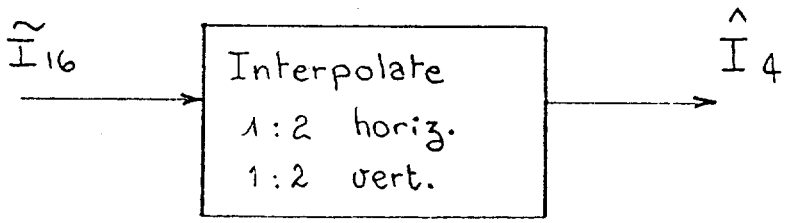


Similarly, we can produce an image subsampled 2:1 in both directions for an overall 4:1 pel reduction. We call this image I_4 :

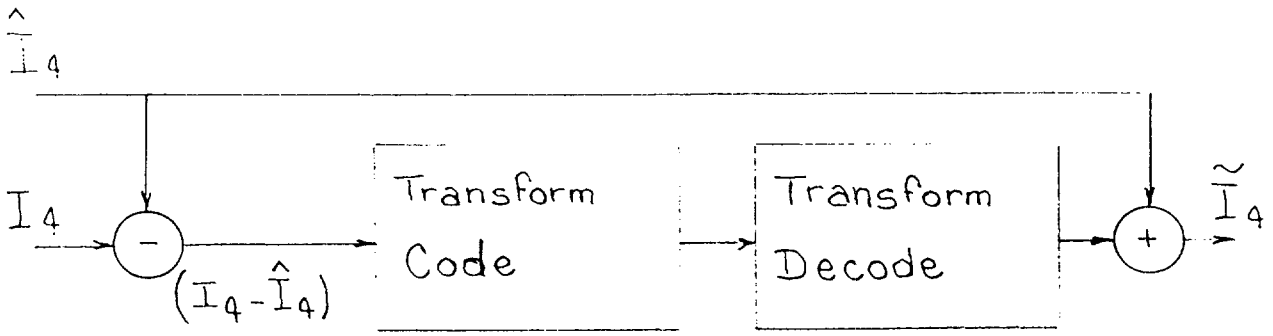
Alternatively, I_4 could be filtered and subsampled by an additional 2:1 in both directions, to give I_{16}.

In any event, I_{16} is then coded by the original ISO – ADCT (n502) coding scheme with high quality at say 1.3 bits per pel. Since there are 16 times fewer pels, the net bit – rate is 1.3/16 = 0.081 bit per original pel. The decoded image is $\tilde{I}_{16}$.

This decoded/subsampled image can be upsampled 1:2 by interpolation in both dimensions to give $\hat{I}_4$, which acts as a prediction for coding I_4. $\hat{I}_4$ can be further upsampled 1:2 in both dimensions to give a full sized image for display. Simple bilinear interpolation is sufficient for most imaging applications, especially taking into account that interpolated values will be automatically corrected at latter stages in the coding.

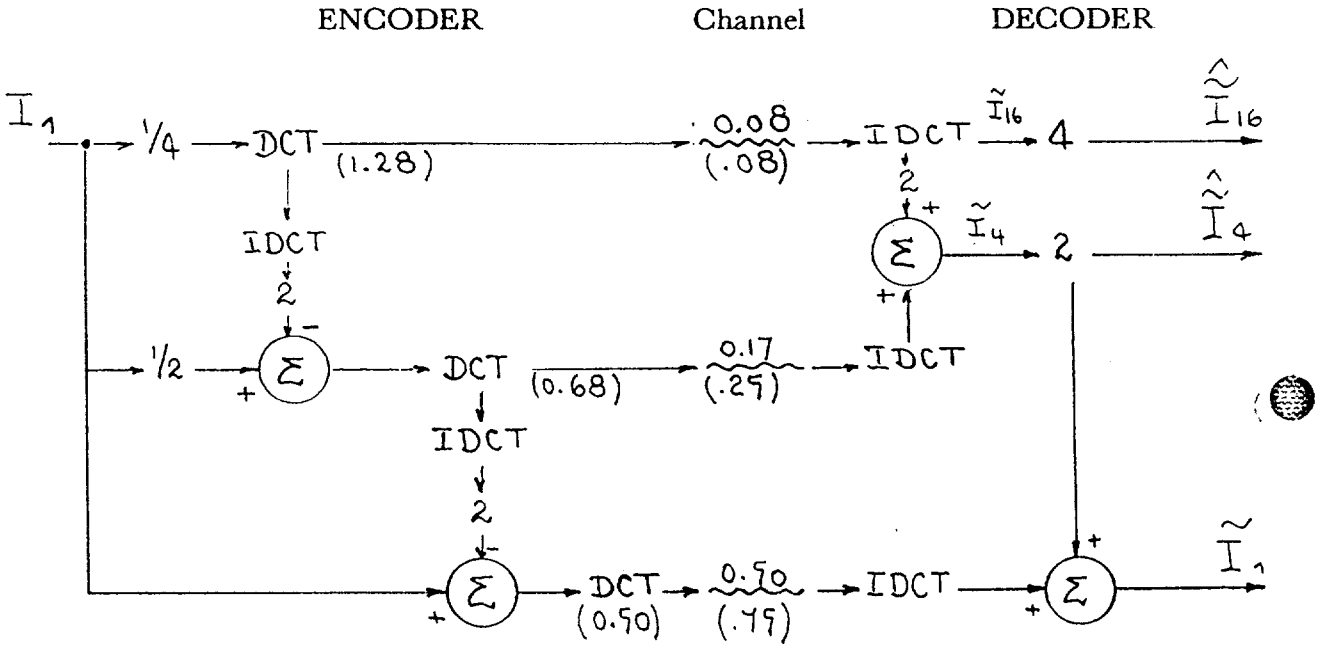


Following this, the residual prediction – error picture $I_4 - \hat{I}_4$ can be transform coded. Note that $\hat{I}_4$ is available at the receiver too. So the decoded version $\tilde{I}_4$ is obtained from :



As before, $\tilde{I}_4$ can be interpolated 1:2 in both dimensions to give $\hat{I}_1$ for display and prediction. If $(I_4 - \hat{I}_4)$ is coded with high quality at about 0.67 bit per pel, then since there are 4 times fewer pels the overall bit-rate is 0.67/4 = 0.167 bit per pel. So the total to achieve this second stage display is 0.081 + 0.167 = 0.25 bit per pel.

Similarly, coding the residual error image $I_1 - \hat{I}_1$ at high quality at about 0.5 bit per pel enables a third stage display $\tilde{I}_1$ to be obtained within 0.75 bit per total bit-rate.



note : "DCT" and "IDCT" = ISO n502

Figure 1 – Summary of the Principles of the Hierarchical Structure

1.3.1.3 Coder Block Diagram

The Progressive Transform Coder (fig. 1) takes in video and stores the frame to be coded in Frame Store 1. Next the image is filtered and subsampled by modules 2 and 3 to produce the first signal to be sent (I_{16} in the above example). At this time Frame Store 7 contains no information, and its outputs is zero. Thus, I_{16} passes through Substractor 4 unchanged to DCT Coder 5, which transforms and quantizes the coefficients that are to be transmitted.

The quantized coefficients then pass to variable world – length Coder 6, which produces bits for the channel, as well as to inverse DCT Decoder 9, which produces decoded pel values. Delay 8 is a short delay needed to match that of the forward and inverse DCT modules 5 and 9. Since Frame Store 7 is output – ting zeros, so is Delay 8. Thus, the decoded pels pass through Adder 11 un – changed to the bilinear 1:2 Interpolator 10, which upsamples the decoded image, quadrupling the number of pels. This interpolated image ($\hat{I}_4$ in the above example) then passes to Frame Store 7 to be used as a prediction in the next stage of progressive transmission.

In the next stage, the original picture is filtered and subsampled by modules 2 and 3 to produce I_4 at the "+ input" of Substractor 4. However, during this stage, Frame Store 7 outputs $\hat{I}_4$, and thus substractor 4 outputs the differential signal $I_4 - \hat{I}_4$, which is then coded , sent and decoded respectively, by modules 5, 6 and 9.

Now, the output of the Delay 8 is $\hat{I}_4$, which is added to the decoded differ – ential signal $\tilde{I}_4 - \hat{I}_4$ at the output of DCT Decoder 9 to produce the decoded signal $\tilde{I}_4$ at the output of Adder 11. This then passes to Interpolator 10 where it is upsampled 1:2 to produce $\hat{I}_1$, which is then stored in Frame Store 7 to be used in the next coding stage.

In the final stage of coding, the original signal I_1 passes unchanged through modules 2 and 3. The prediction $\hat{I}_1$ from Frame Store 7 is substracted, and the differential signal $I_1 - \hat{I}_1$ is DCT coded and sent as above, thus completing the transmission of the picture.

1.3.1.4 Decoder Block Diagram

The Progressive Transform Decoder (fig. 2) is essentially the inverse of the coder. Initially, Frame Store 17 contains nothing and outputs zeros.

At the first stage, the received quantized transform coefficients (I_{16} in the above example) are produced from the the received channel bits by variable world – length Decoder 13. These then pass to inverse DCT Decoder 16, which produces decoded pels values. Since Frame Store 17 is outputting zeros, so is Interpolator 14. Thus, the decoded pels $\tilde{I}_{16}$ pass through Adder 15 unchanged to the Frame Store 17.

Frame Store 17 has two outputs, which operates at different rates. Output B operates at the display rate and passes to programmable interpolator 12, which produces the display picture. During the first stage, Interpolator 12 upsamples I_{16} by 1:4 in both dimensions. During the next stage of progressive transmission, output A of Frame Store 17 passes to the bilinear 1:2 Interpolator 14, which upsamples the previously decoded image (I_{16} here), quadrupling the number of pels. This interpolated image ($\hat{I}_4$ here) then passes to Adder 15. During this stage the Decoders 13 and 16 produce the decoded differential signal $\tilde{I}_4 - \hat{I}_4$, which is added to $\hat{I}_4$ to produce the decoded signal $\tilde{I}_4$ at the output of Adder 15. $\tilde{I}_4$ then passes into Frame Store 17.

Note that the programmable interpolator 12, allows for other upsampling ratio than 1:2 or 1:4, as described herein, and thus, the decoder could adapt the input – resolution to the display resolution.

In a final stage of coding, Decoders 13 and 16 produce the decoded differential image $\tilde{I}_1 - \hat{I}_1$, Interpolator 14 produces $\hat{I}_1$, and Adder 15 outputs a final decoded image $\tilde{I}_1$, thus completing the transmission of the picture.

Note that the final coding stage could be repeated if we desired to improve the quality of the transmitted picture. To do so, we simply bypass the bilinear interpolators 10 and 14 , and perform another coding of the residual error image $I_1 - \tilde{I}_1$ .This correction signal is then added to the previously received picture, thus improving its quality.

Furthermore REVERSIBILITY, will simply be obtained by simply bypass – ing the DCT Coder and Decoders 5, 9 and 16, then original pel values can be coded, and LOSSLESS entropy coding can be carried out by Encoder 6.

Block Diagrams for the progressive transform Coder and Decoder

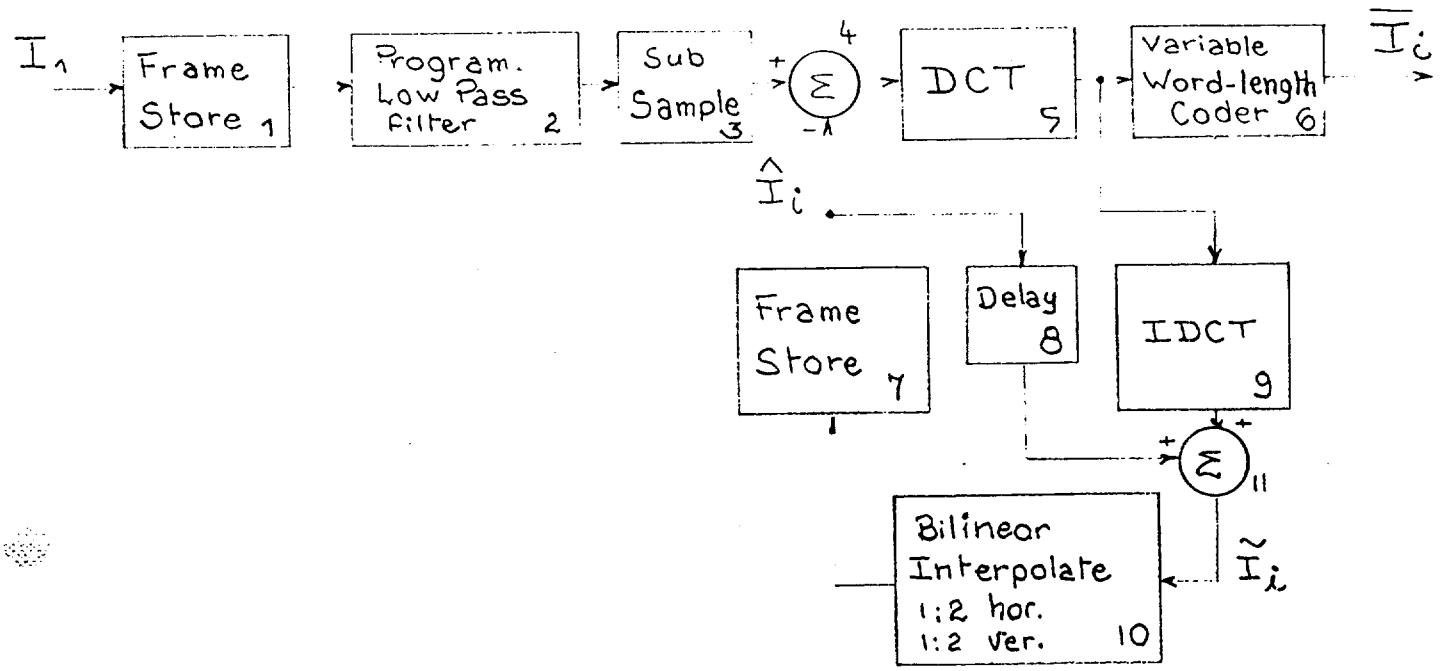


Figure 2 – HIERARCHICAL TRANSFORM CODER

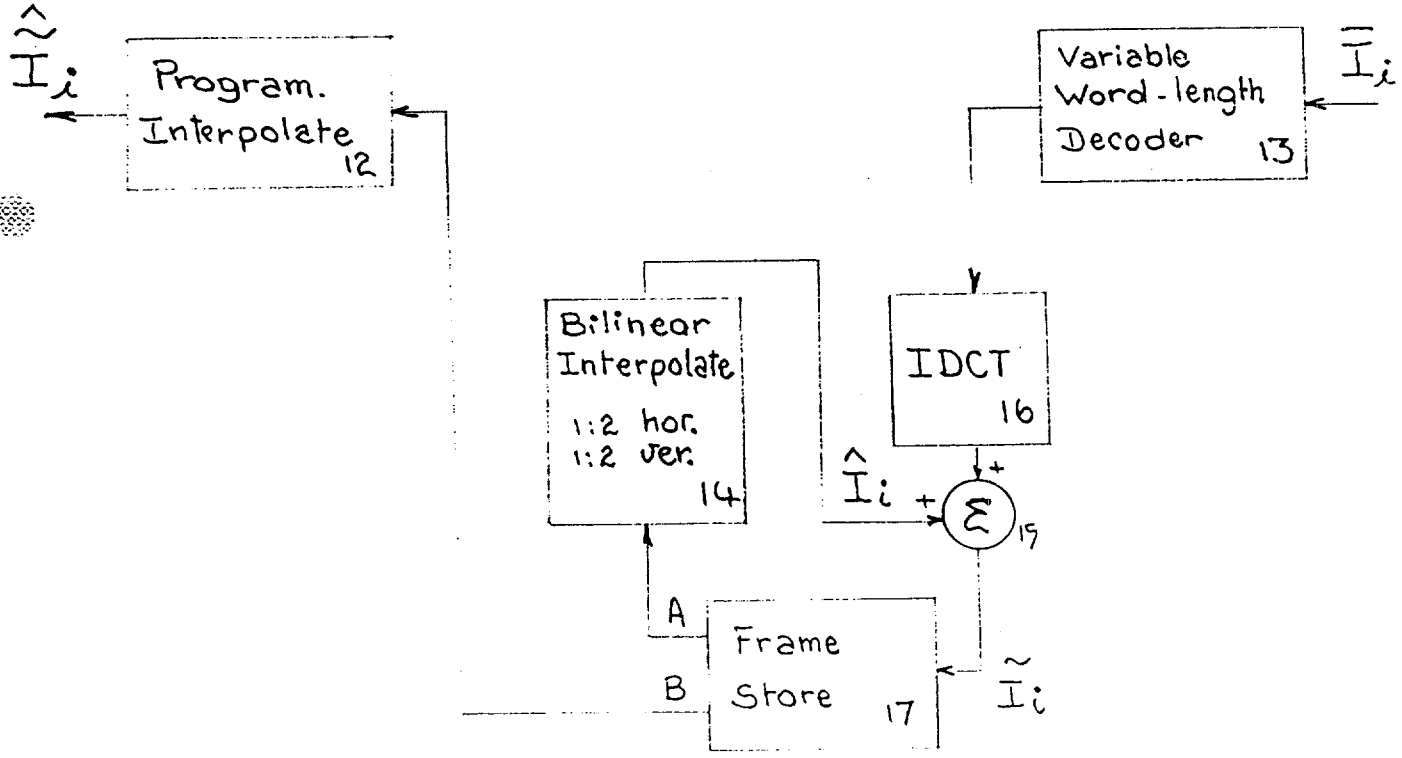


Figure 3 – HIERARCHICAL TRANSFORM DECODER

Figure 4

DCT CODER

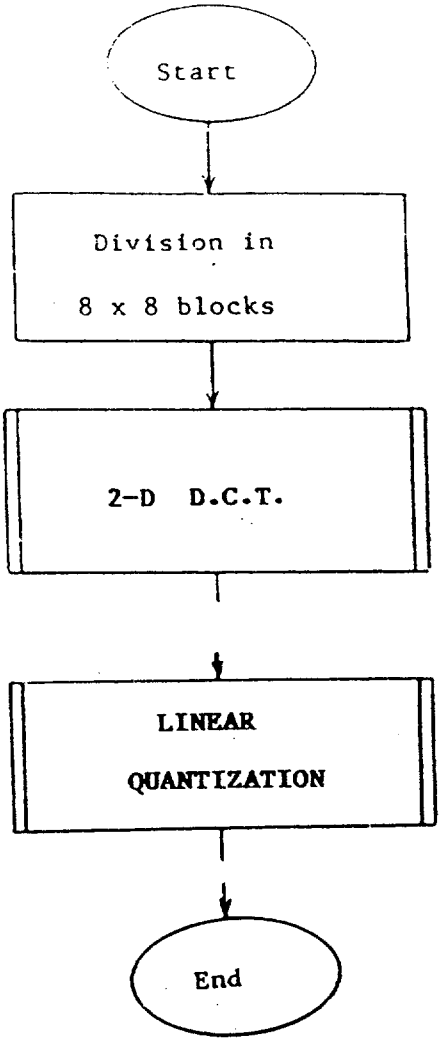
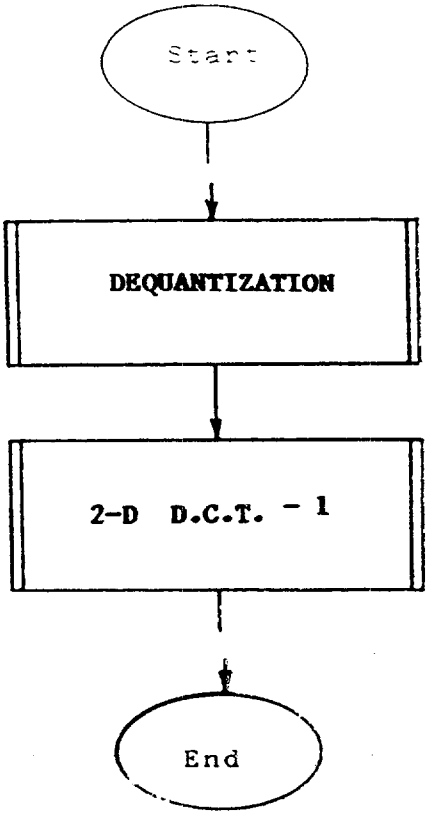


Figure 5

Inverse – DCT DECODER



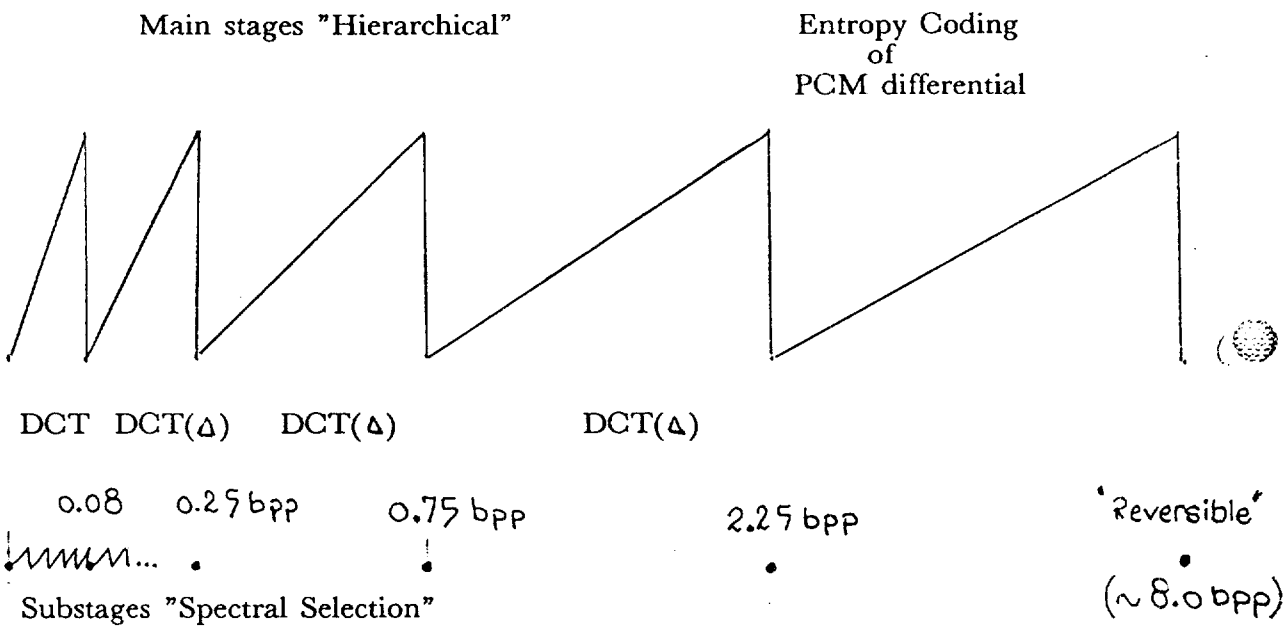
1.3.2 Coding Structure for sequential update

As aforementionned, Sequential update can be carried out in two ways. One is built on the "Progressive Hierarchical Structure", the other is built in a "one – pass" coding structure. The latter, is by far the most "natural" for a block coding technique, and requires no transmission buffer, unlike the former that imposes to resequence the data before transmission.

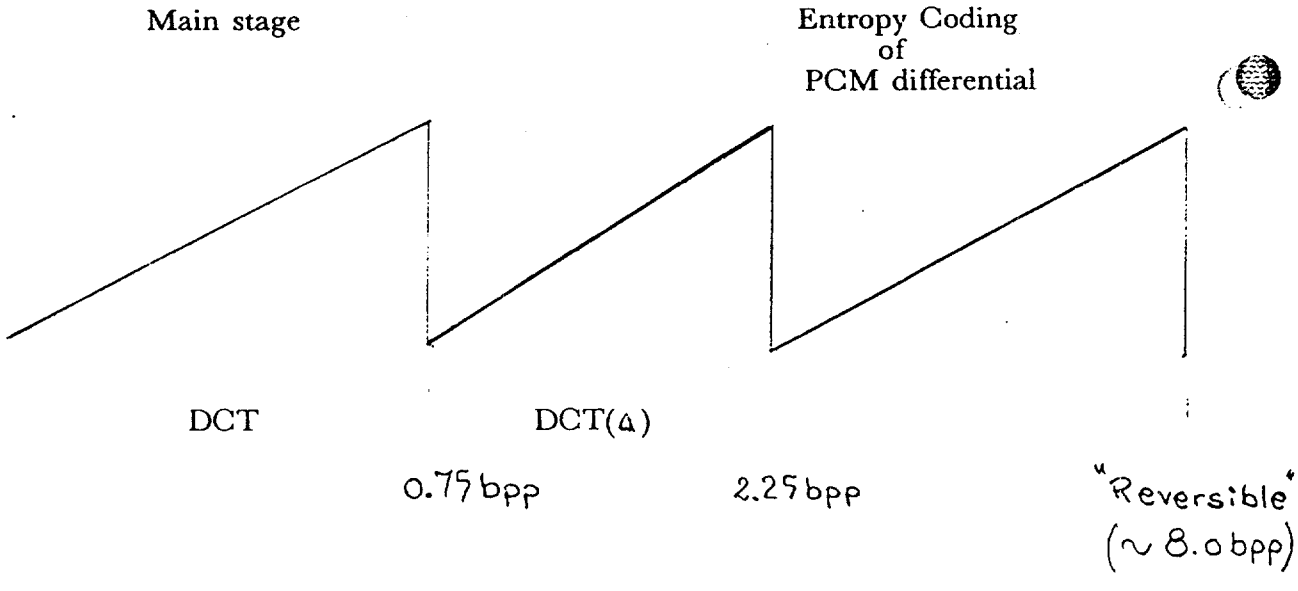
The drawback of the simplest one, beeing that it does not fully match the picture coded in progressive update at the same coding rate. But, does that mind the service????

Progressive and Sequential Schemes

PROGRESSIVE



SEQUENTIAL



NOTE that the first Main stage of progressive update may be equivalent to main stage of the sequential update

(*) Note that sequential might be changed before the january meeting, as agreed in n599.

General

1.3.3 Technical Description

1.3.3.1 Filtering – Subsampling – Interpolating

See Tables 4 , 5 , 6 and 7 in Annex A.

1.3.3.2 Transformation

A 2 – dimensional Discrete Cosine Transform is applied to 8 x 8 blocks of pixels.

1.3.3.3 Linear Quantization

After transformation, the coefficients are quantized linearly, in using a quantization step dependent on the spectral position of the coefficient to be quantized. The quantization process is simply a round – off operation on the co – efficient value divided by the quantization step.

NOMINAL quantization steps are given by the quantization matrices (ta – ble...) which have been derived from a psychovisual experiment. In other terms, the quantization steps have been set to be the "Just Noticeable Ampli – tude Perception" at normal viewing distance, of each coefficient considered in – dependent. Those values used directly will lead to a very good picture quality at a coding rate < 1.0 bit per pel, but by scaling them with a common factor, in the general case, we can adjust the bit rate and correspondingly the picture quality, to nearly any desired value.

The values shown are those presently used for pictures described according to CCIR A – 601, mode "4 – 2 – 2" – as required for ISO testing purposes –, but as they are part of the overhead (= 128 bytes) they can be changed for other picture resolutions or specific applications (Graphic arts, Synthetic images,...). However, an example of quantization matrices for resolution conforming to CCIR A – 601, mode "2 – 1 – 1" is given. Obviously, this aspect needs future re – finements, but the coding structure is fully open.

1.3.3.4 Adaptive Entropy Coding

After quantization the transform – quantized block is sparse : only few AC coefficients are non – zero.

The content of the block is transmitted by sending information on the value and the position of the coefficients, and for this purpose the transform – quantized block is described by a 1 – dimensional scan, that is the simple "zig – zag" scan, from upper left hand corner (table...)

The DC coefficients are treated seperatly, they are represented in DPCM and encoded in using particular 1 – dimensional VLC tables.

AC coefficient relative positions and number of bits to represent subsequent AC value are encoded using 2 – dimensional VLC tables. The AC value itself is coded in PCM with the corresponding number of bits expressed by the previous code.

The VLC tables may be predefined or customized to specific image(s) or application(s). In the latter, they are part of the overhead and they account for about 400 bytes in total. In any case, the recommendation would defined a standard set of predefined VLC tables that could be used as default values, as long as no particular ones are transmitted.

The present description will still be based on the full adaptive case – i.e. customized VLCs –, owing to the tight ISO schedule it was impossible to derive the faithfull predefined set, BUT proofs have been made that predefined VLCs are definitly usable with nearly NO degradation of the coding rate.

2.0 Coding Algorithm

2.1 FUNCTIONALITIES

2.1.1 Exchange of Parameters

Header of general parameters : Resolution, nb of stage, sequential or progressive,....This will be defined latter within JPEG.

Header for dynamic redefinition of variables : VLCs, thresholding matrices,... Some are coded in the simplified syntax (par. 2.2.2.), others will be defined latter within JPEG.

2.1.2 Number of Stages

The available number of stages is in fact very large in regard to the actual need.

However, those stages can be splitted in main stages and in sub-stages, as follows:

- The main stages are the hierarchical ones - 9 in total, (- 1,2 and 4 sub-sampling in both dimensions), the extra ones based on iterations of the DCT on the decoded errors of the full resolution image - this number is not fixed, it may be 5 or 10 in total -, and lastly the reversible scan, in 1 or 2 main scans.

- The substages are based on spectral selection of the coefficients according to a predefined scheme (e.g. one coefficient per block at a time), for each main stage, excepted the last PCM entropy coded reversible scan that may be divided in a large number of subsampled scans.

In total, the coding structure allows for up to one hundred scans !!!!

For edition purpose, in progressive mode, the user will simply define the desired number of stages, and then dedicated routines - not yet defined, because largely dependent on the service - will automatically route the image data in the processing structure that produces the most pleasing display.

This point - typically service dependent - is clearly for further study, but in any case, the coding structure does not impose any constraint to the user or the service.

2.1.3 Resolution Control

The only parameter that is sensitive to picture resolution, is the quantization matrix (Y,Cr,Cb). However, note that we might, very likely, specify an "aver - age" matrix that could give very good results in a variety of situations, but at the expense of loss of excellence.

Nevertheless, herein (Annex...) are described the to - day best quantizing matrices for Y,Cr,Cb conforming CCIR - A601, mode "4 - 2 - 2", as required for psychovisual selection purposes by ISO. Furthermore, Quantizing matrices for Y,Cr,Cb conforming CCIR - A601, mode "2 - 1 - 1", are proposed but have not been tested yet.

Finally, it is considered that this aspect is fully THEORITICALLY solved, but practically, partially answered. Indeed, it is always possible to define a ded - icated quantization matrix to fit a specific resolution (or application) and at a cost of, only, 128 bytes (!) for transmission to the receiver. For typical resol - utions or applications, average or default quantization matrices, would be stored.

For the user, in any case, it will simply consist in specifying the resolution of the image to be coded, that is a mandatory parameter of the syntax for any image communication services.

2.1.4 Level Range Control

This coding algorithm has been typically designed for 8 bit per component image data. However, its applicability to : 1 bit.....7 bit per component image data, is straightforward by simple normalization of the input value and, further - more, by eventually the use of the adaptive entropy coding option.

Above 8 bit per component image data, the present scheme has to be revised, not on the principles, but on the implementation side (e.g. DSP chip with 32 bit internal resolution instead of the consumer DCT chip) and the adaptive entropy coding has to be adapted (i.e. the number of events is increased).

2.1.5 Quality Control

This aspect is exclusively controlled by the quantization matrices. The doc - umentation proposes NOMINAL quantization matrices that give TYPICALLY and CONSTANTLY, for any kind of images, an excellent psychovisual picture quality.

To modify this nominal picture quality, the user has simply to scale uni - formly the quantization matrices with a common factor, that has to be part of the overhead syntax. To that end, the user would be provided with a picture quality scale that automatically would convert it, to a scaling factor.

However, future refinements of this simple control are fully open, via the definition of dedicated quantization matrices, that may be part of the overhead syntax.

2.1.6 Bit Rate Control

The bit rate is also, exclusively, controlled via the quantization matrices. However, unlike quality control, the bit rate control depends on the picture complexity and, since the scheme is adaptive, two pictures of equivalent psychovisual quality, will not, in general, be coded with the same total bit. So, in order to ensure maximum subjective quality of a given main stage, the mini – mum threshold factor F giving a bit rate below or equal to the desired bit rate for that stage is calculated.

The algorithm is simple :

° the relation between the bit rate B and the threshold factor F has been empirically determined and is rather well represented by the expression :

$$\log(B) = a*\log(F) + b \qquad (1)$$

° A subroutine giving B as a function of F has been implemented. It is based on a Newton – Raphson iteration that gives, in a few steps, the desired value of F.

So, if a very precise bit rate is required, two solutions are offered : i –) iter – ations of the coding process to converge to it or ii –) coding of the image in progressive update to just above the desired bit rate and then, transmission of the exact amount of the compressed image data.

2.1.7 Synchronous Operability

The coding/decoding structure is fully in synchronous operation for :

° the sequential update

° the progressive update with a sufficient number of main stages that are basically each in sequential format

The coding/decoding structure is nearly in synchronous operation for :

° the progressive update as soon as sub – stages of spectral selection are used

in this latter case, the synchronous operability is perceptively met (i.e.the real time constraint is fully met), but not conceptually.

Another aspect to mention, is the use of the adaptive entropy coding option, where also, the synchronous operability is conceptually not met, since pre – processing of the image is needed, before transmission.

Nevertheless, for applications on ISDN, the mentionned pre – processing times are always insignificant in regard to the transmission time – in other terms, the processors are fast enough to conceal this aspect. For broad – band channels, progressive update may be no longer needed and so, full synchronous operation is available.

2.1.8 Reversibility

The DCT is known to be difficultly reversible. So, for reversibility, the coding structure, basically, escapes from DCT to PCM, and simply makes the differences between the DCT decoded image – at one appropriate coding rate – and the original, and then code them via a lossless coding scheme – typically, an Huffman coding technique.

A alternative is described in Annex B.

2.1.9 Other Features

The prominent features of this algorithm are :

- Simplicity / efficiency
- Free licensing
- Foreseen, intra – mode coding Compatibility with visiophony at 64 kbit/s (CCITT sg XV), visioconference at 384 kbit/s (CCITT sg XV) and Digital Television (CCIR).

2.2 PROGRESSIVE CODING

2.2.1 Flow Charts and Explanations

CODING AND DECODING FLOW CHART OF DCT-CODING

Notation

bits :array(1..maxcodelength) of integer; indicates the number of huffcodes using a certain codelength

C :input value for code-generation

C1 :input value for code-generation

C2 :input value for code-generation

code :integer

codesize :array(low..high) of integer; indicates the code-length of a certain coefficient

coefsize :function that tells how many bits are required to represent coefficient. Because the number of bits in a coefficient is known on beforehand, the following bit allocation may be used:

input values	nr bits	output values
-1,1	1	0,1
-3,-2,2,3	2	00,01,10,11
-7--4,4-7	3	etc
-5--8,8-15	4	etc
-31--16,16-31	5	etc
-63--32,32-63	6	etc
-127--64,64--127	7	etc

diag :array(0..63) of transformcoefficients

diffdc :one dimensional array consisting of the differential DC-coefficients

distemp :integer

EOB :code meaning End Of Block

freq :frequency-statistic equal to either freqac or freqdc

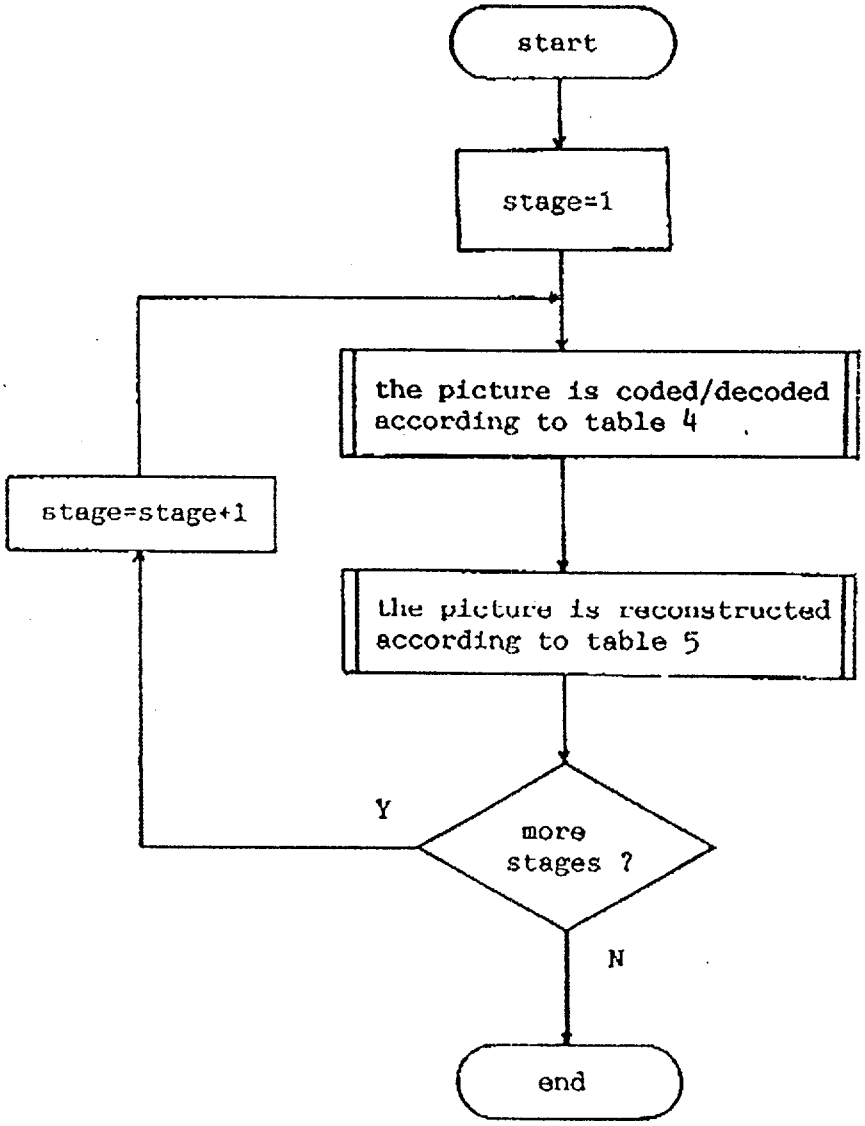
freqac :two dimensional array used for the AC-frequency-statistics. Each entry determines the distance to the next non-zero coefficient along a zig-zag scanning path along with coefsize of the next coefficient

freqdc :one dimensional array used for the DC-frequency-

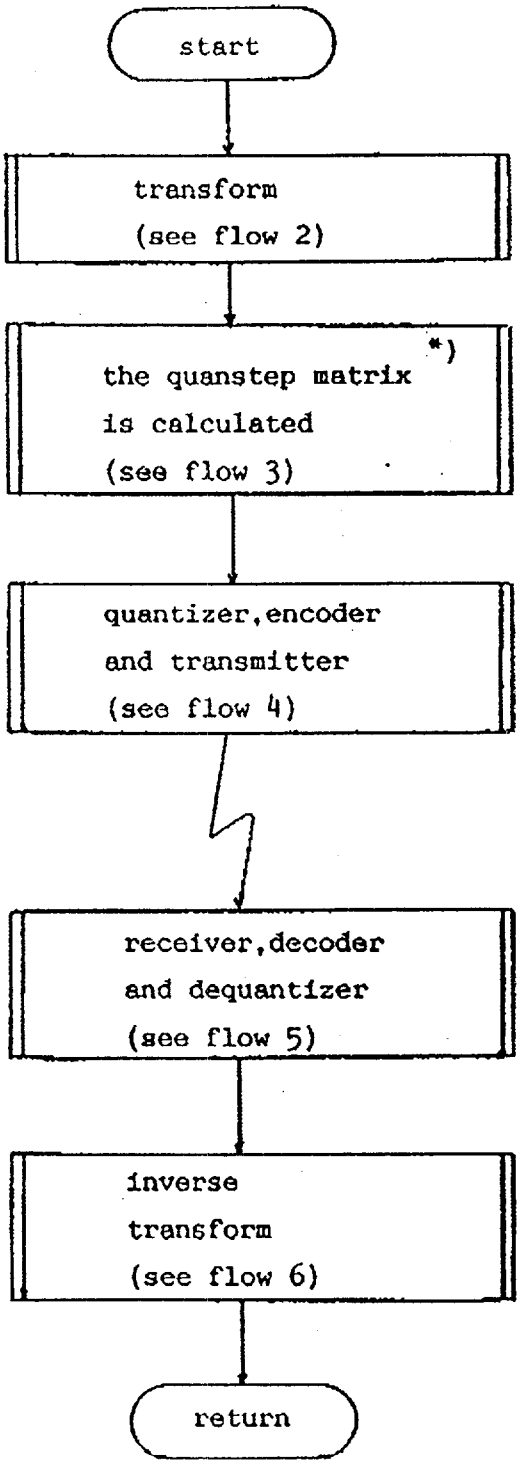
```
                statistics
high            :integer=constant; indicates the highest possible
                 coefficient
huf             :array(low..high) of integer
hufftab         :array(0..numberofcodes-1) of
                          record
                            value    :integer; coefficient value
                            huffcode:integer; coefficient huffcode
                            size     :integer; huffcode size
                          end
i               :integer
j               :integer
k               :integer
low             :integer=constant; indicates the lowest possible
                 coefficient
maxcodelength:integer=constant describing the length of the
                 longest possible huffman code. If all posibilities
                 are >10^-6, which is the case for DC-coefficients
                 as well for AC-coefficients and runlengths then
                 maxcodelength<21
nil             :integer
numberofcodes:integer; describing the number of different
                 values to be encoded

others          :array(low..high) of integer; tells which other
                 codesizes should be incremented
point           :integer
quanstep        :array(0..7,0..7) of integer described in table 1
                 and table 2
run             :integer
si              :integer
stage           :integer; indicates the stage in the progressive
                 update
temp            :integer
tresholdfactor:integer
xfrm            :array(0..7,0..7) of byte consisting of
                 coefficients
zigzag          :array described in table 3
```

Progressive update coding/decoding.

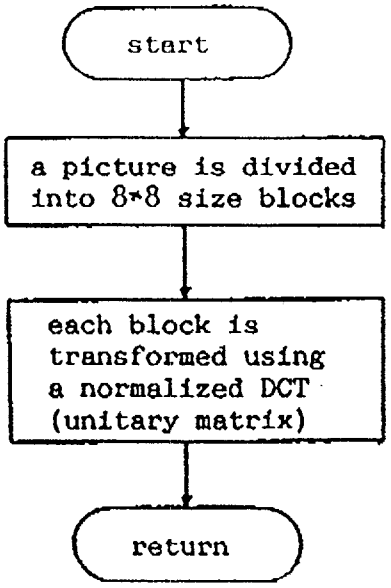


Flow 1 Index of the description of the algorithm

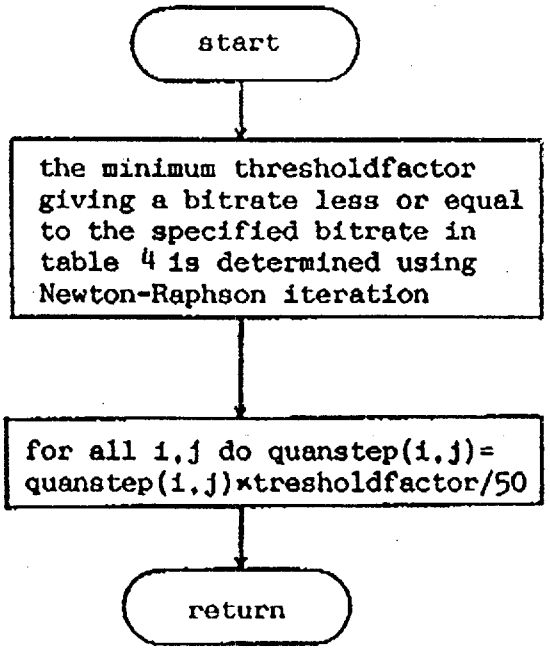


*) this part has only been implemented for ISO selection purposes, and will be removed for real applications

Flow 2 Transform

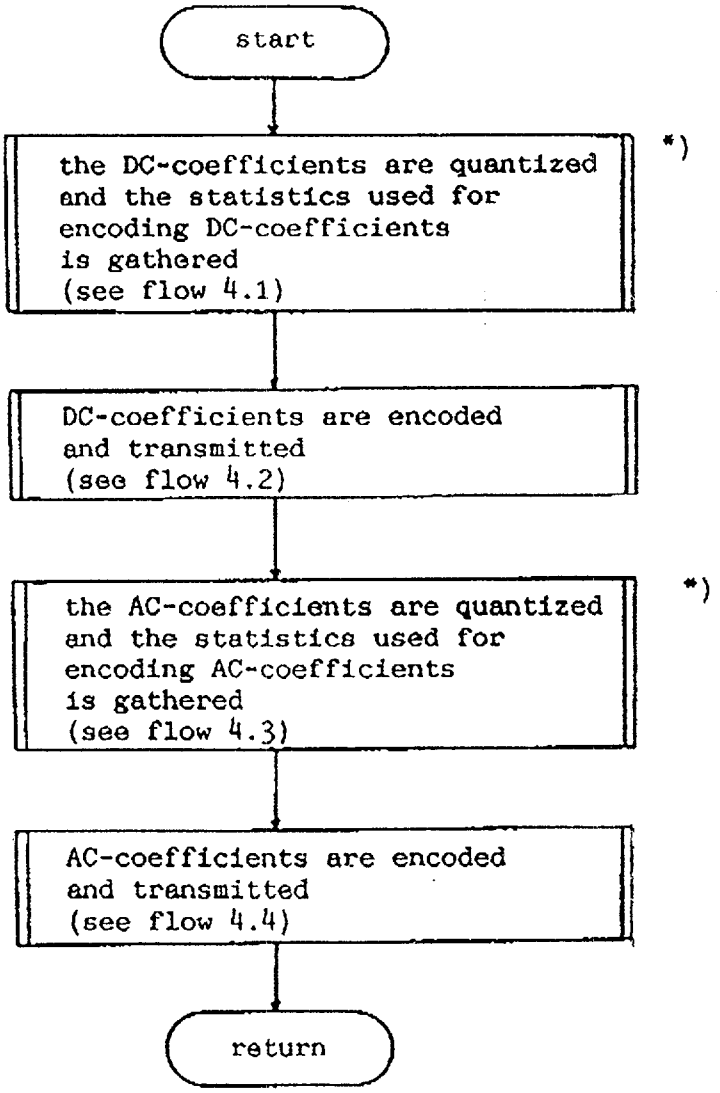


Flow 3 The quanstep matrix is calculated *)



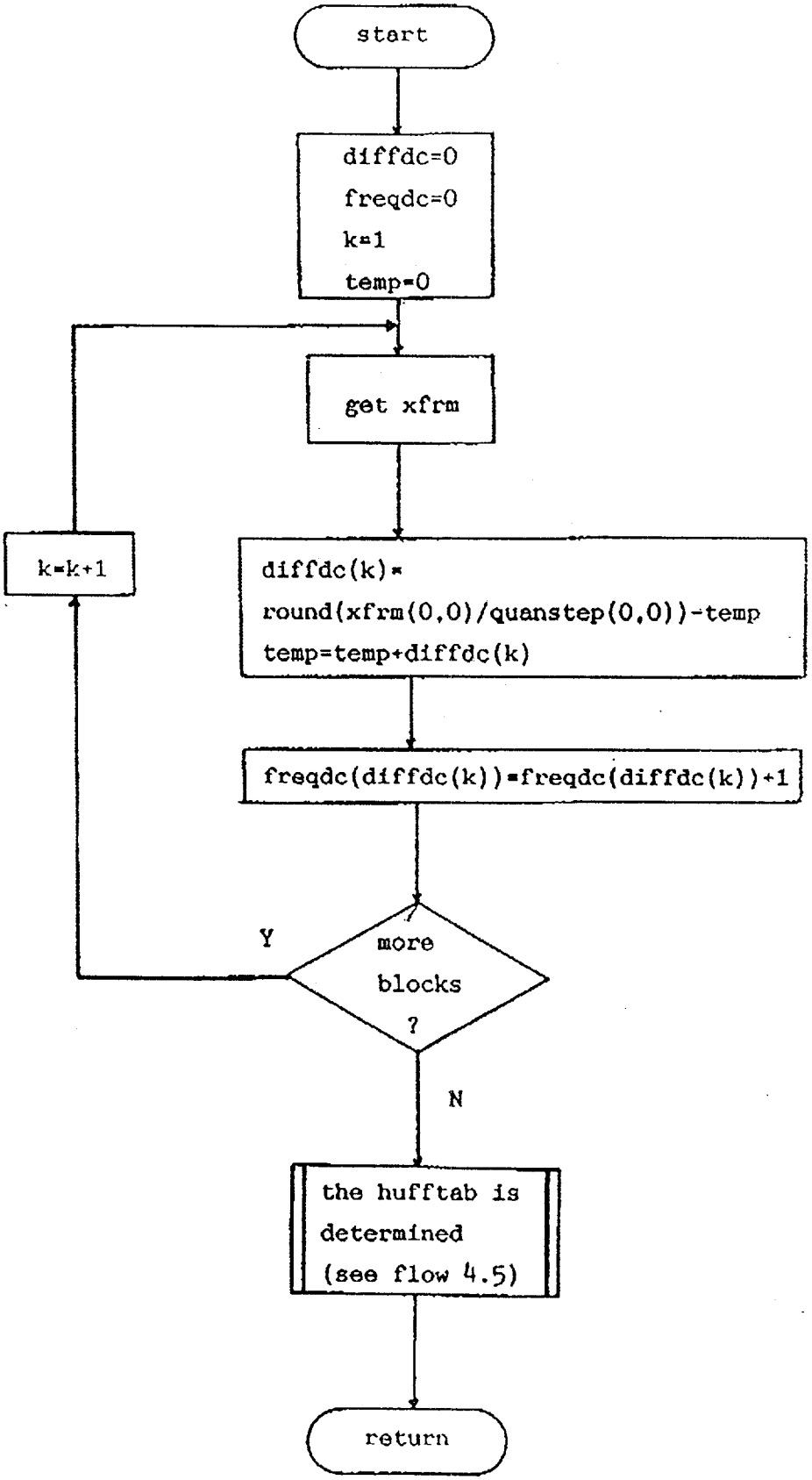
*) this part has only been implemented for ISO selection purposes, and will be removed for real applications

Flow 4 Quantizer, Encoder and Transmitter

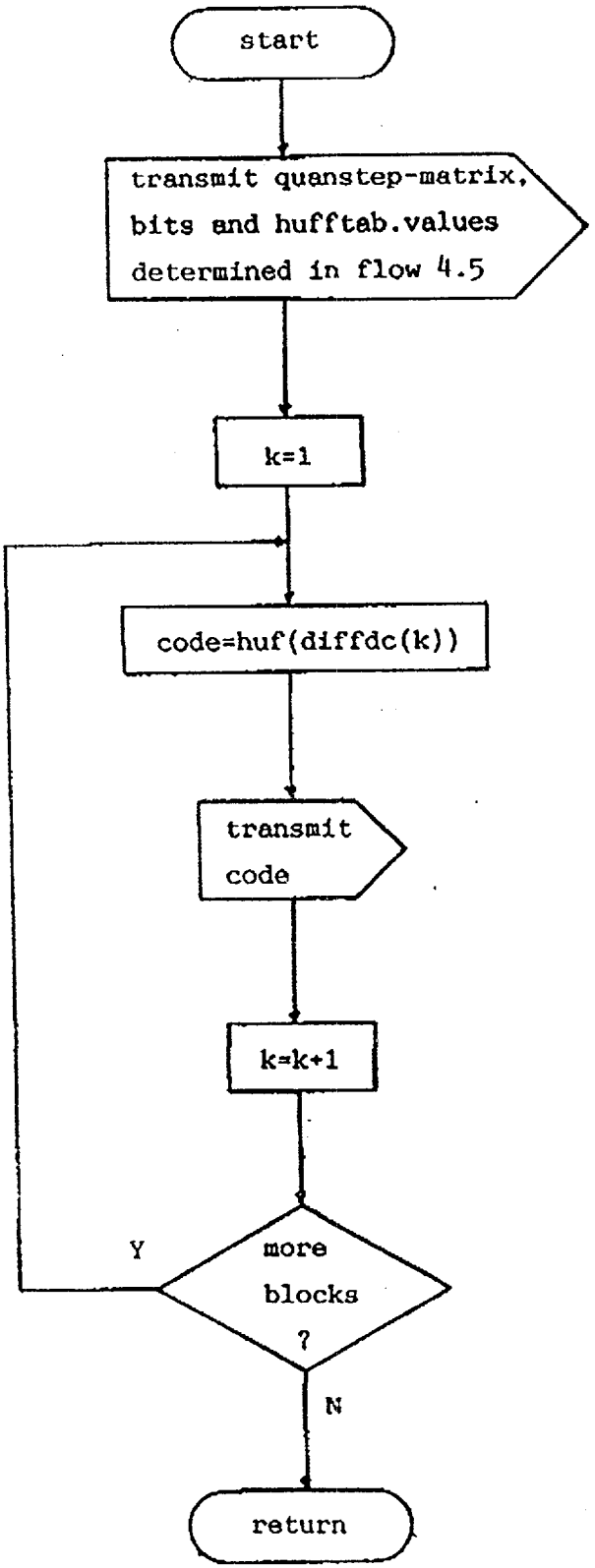


*) this part could be replaced by predefined VLC-tables in real applications

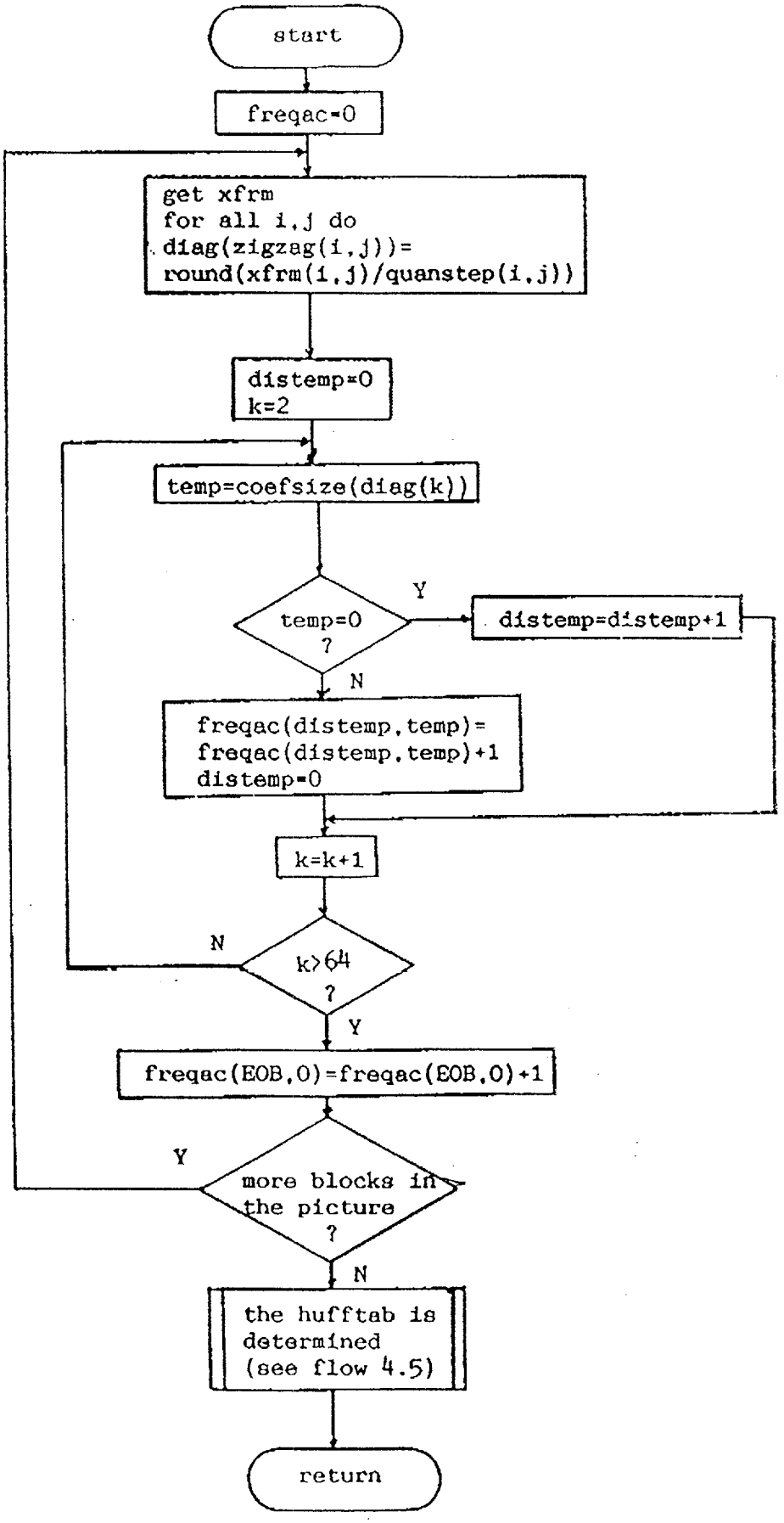
Flow 4.1 Statistic used for DC-encoding



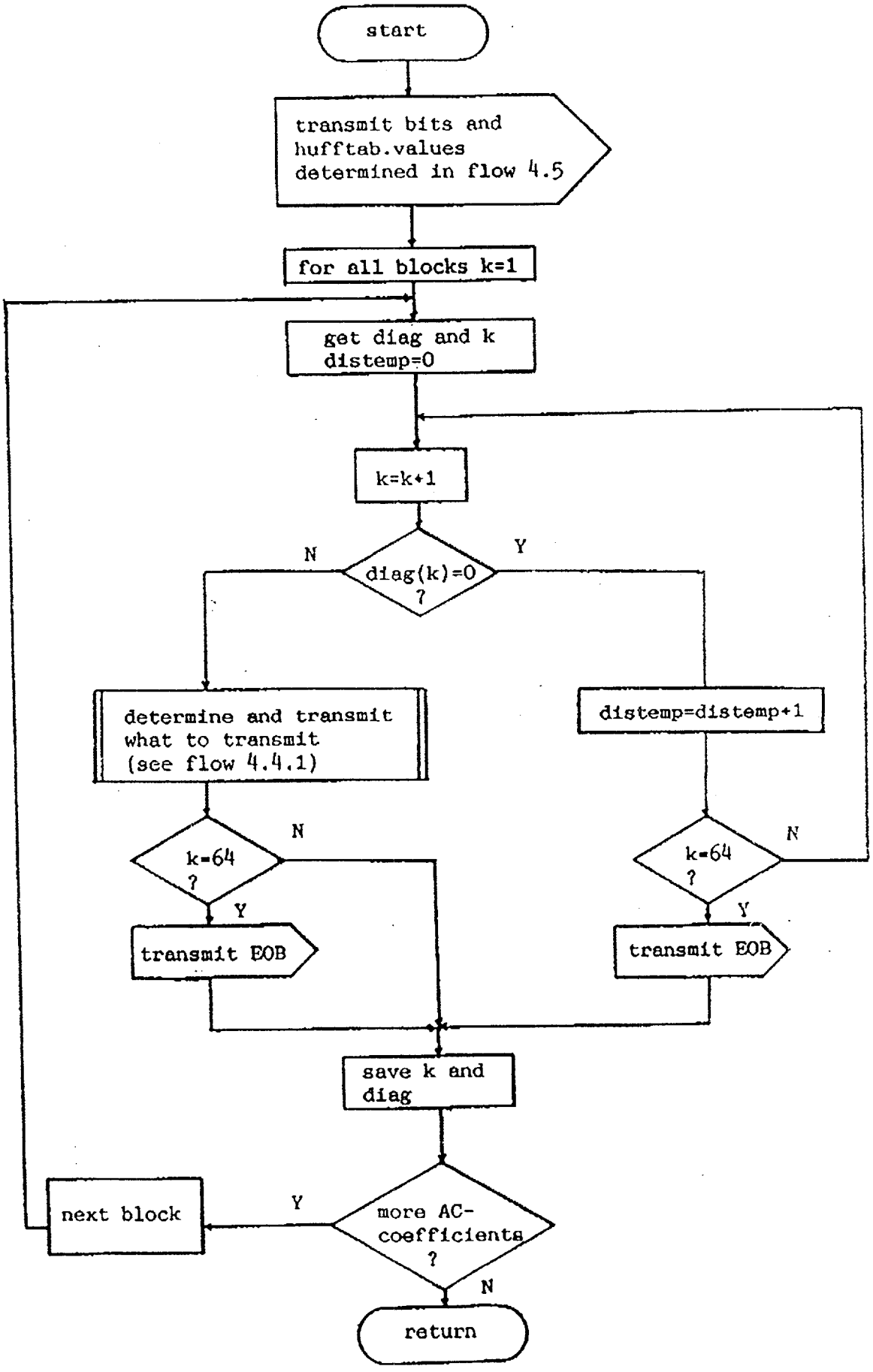
Flow 4.2 DC-coefficients encoding and transmitting



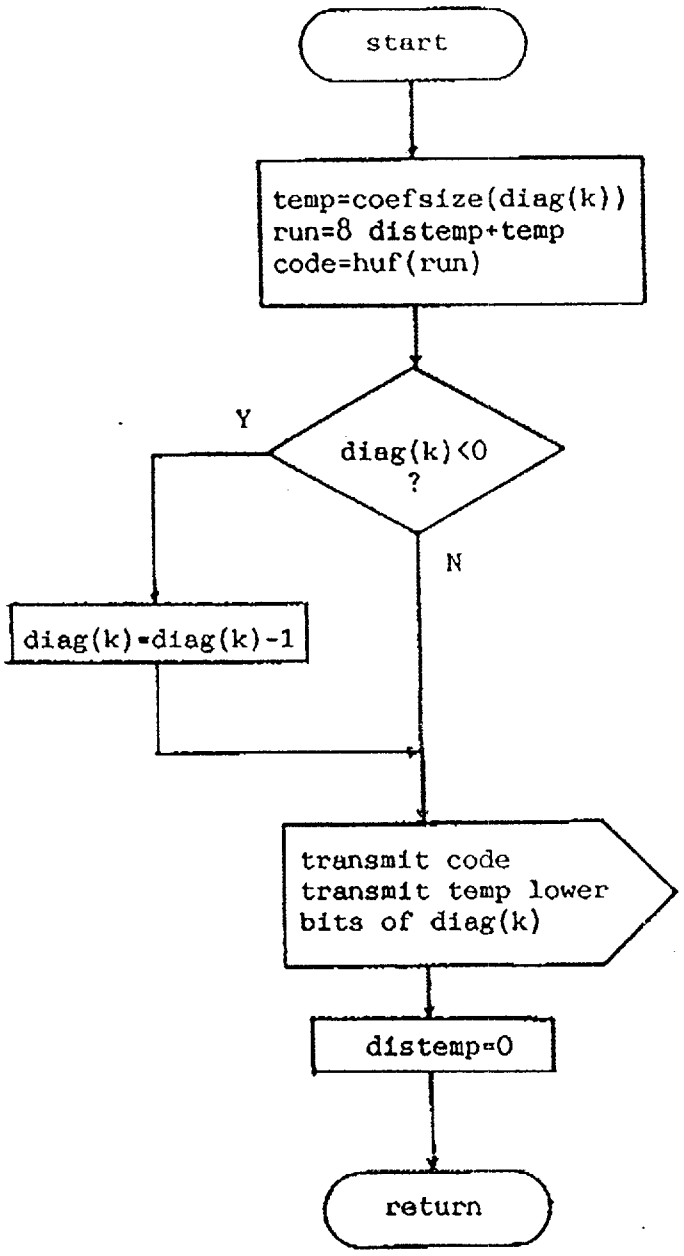
Flow 4.3 **Statistic used by AC-encoding**



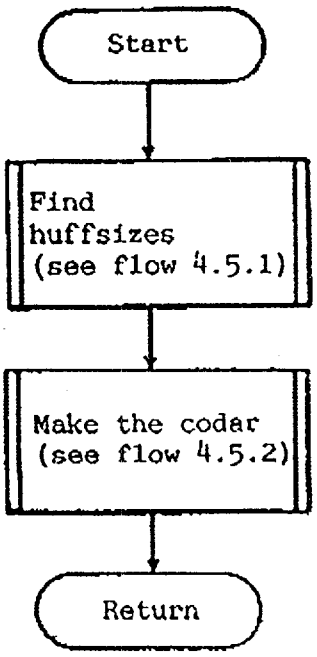
Flow 4.4 AC-encoding and transmitting



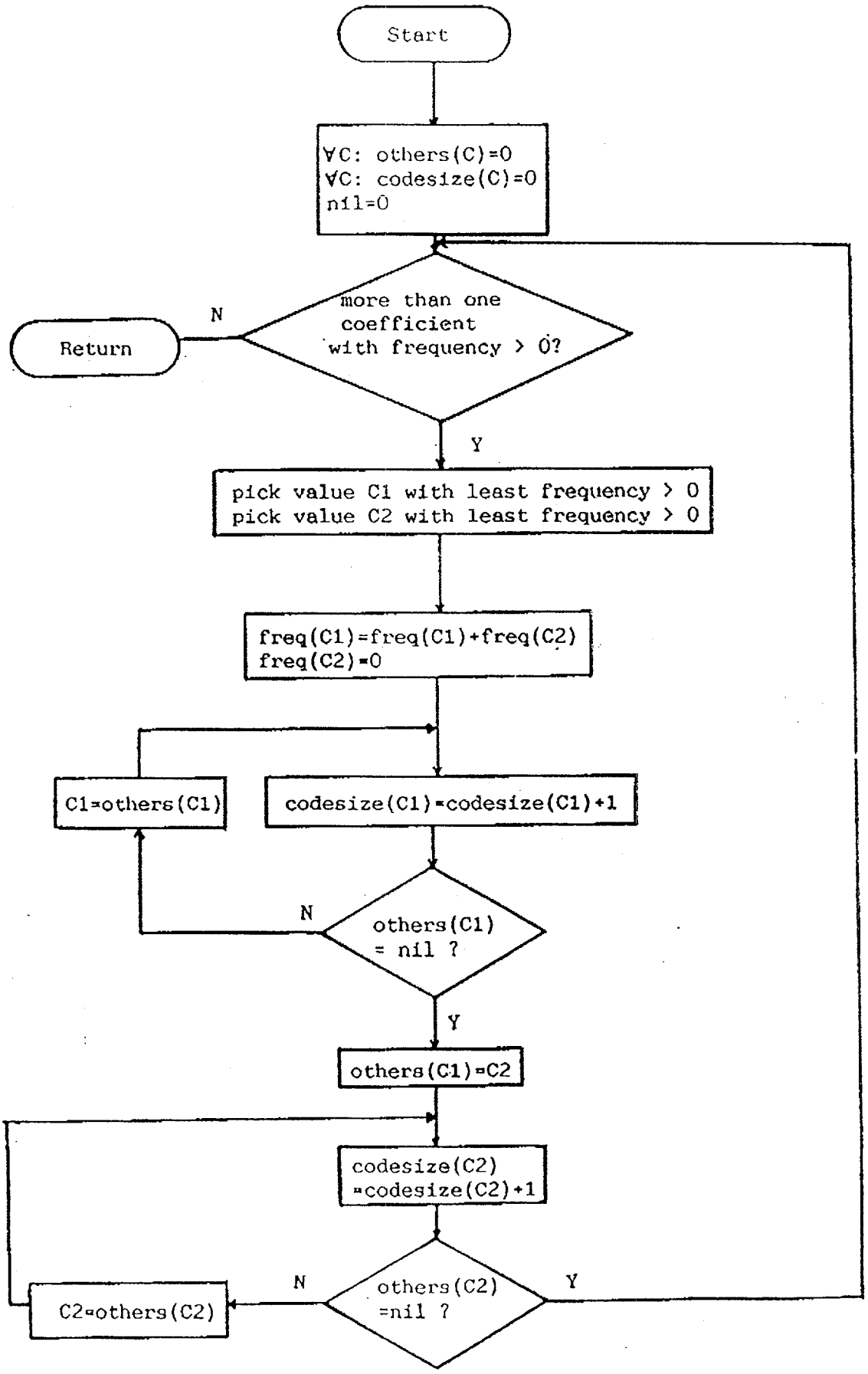
Flow 4.4.1 Determine and transmit what to transmit



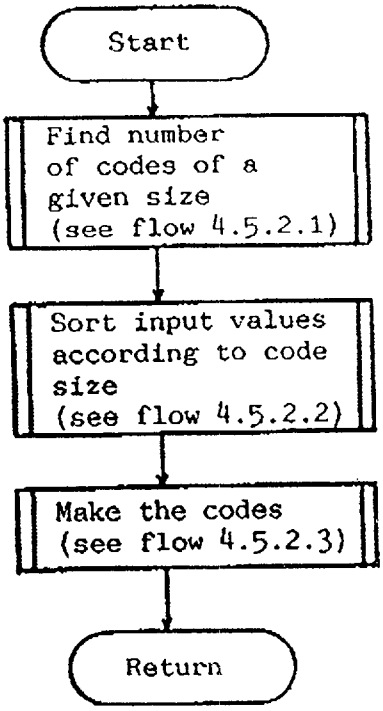
Flow 4.5 **The hufftab is determined**



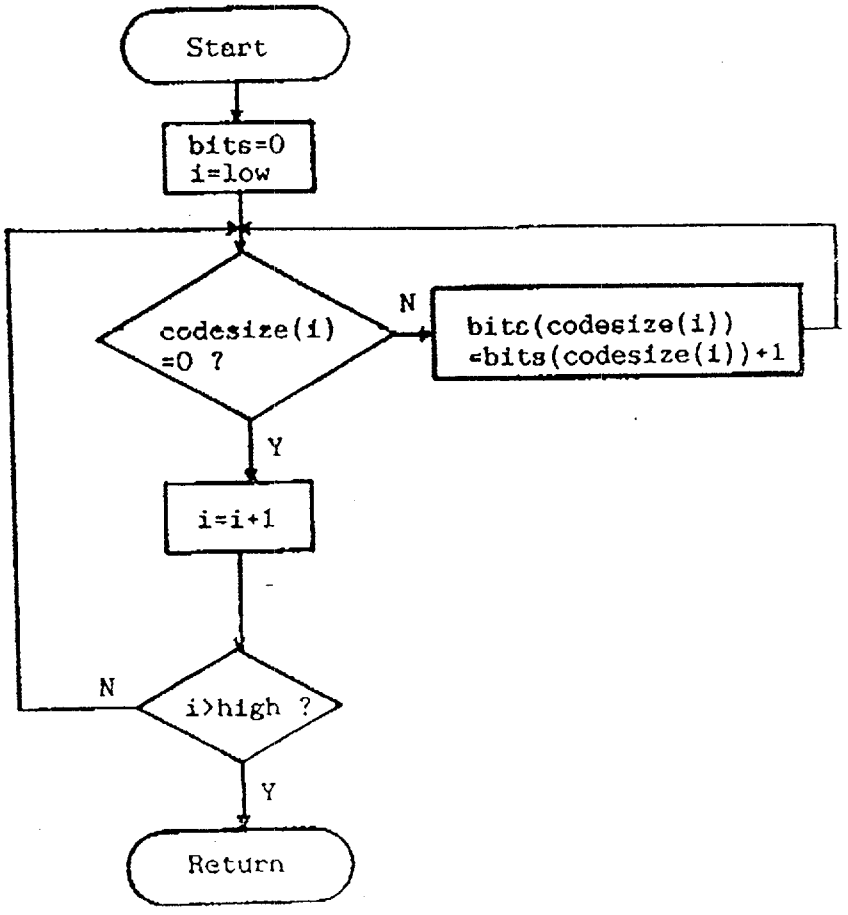
Flow 4.5.1 Find huffsizes



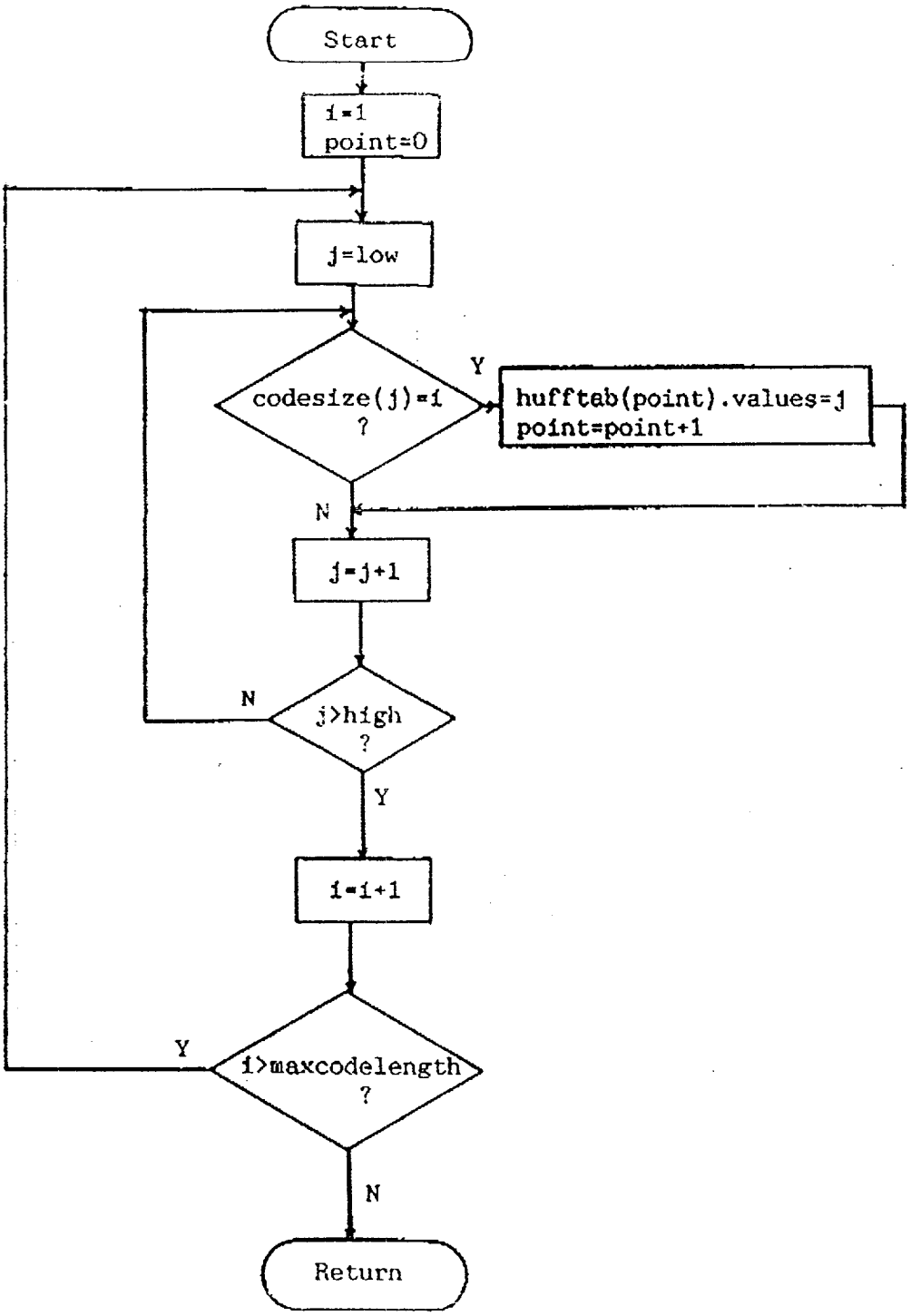
Flow 4.5.2 Make the codar



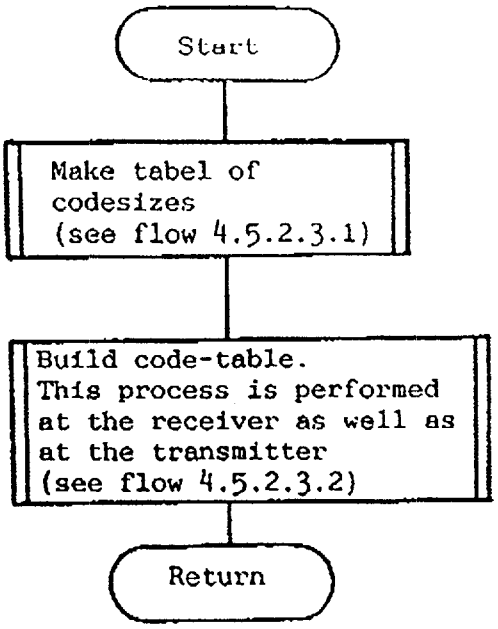
Flow 4.5.2.1 Find number of codes of a given size



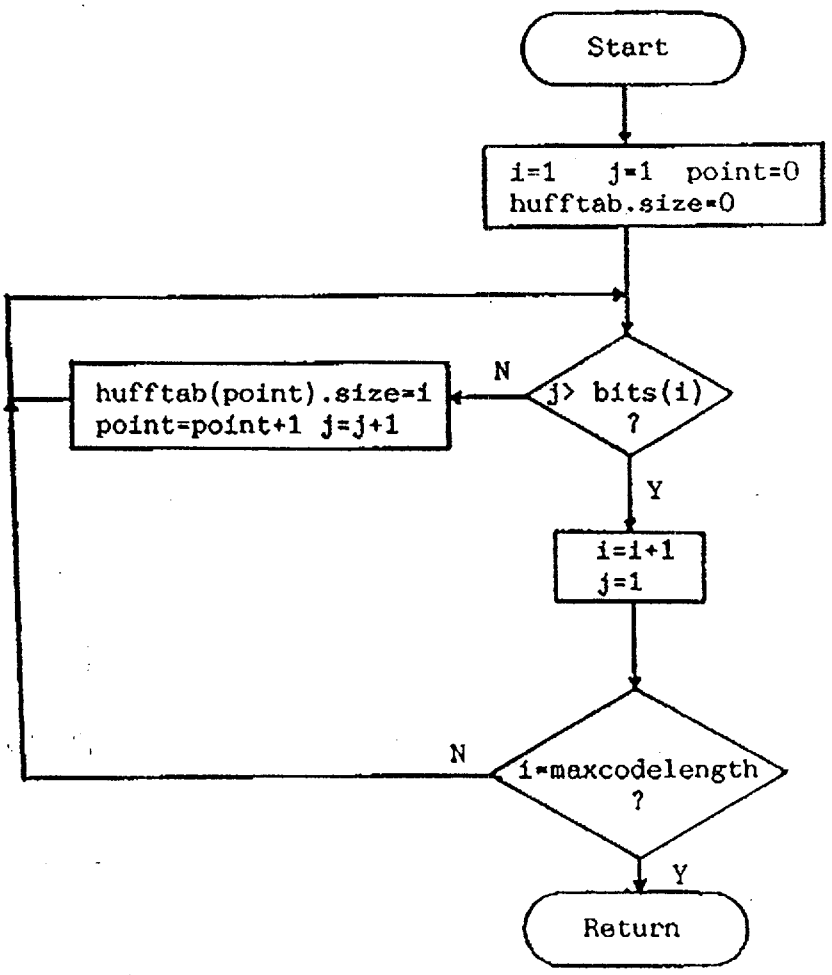
Flow 4.5.2.2 Sort input values according to code size



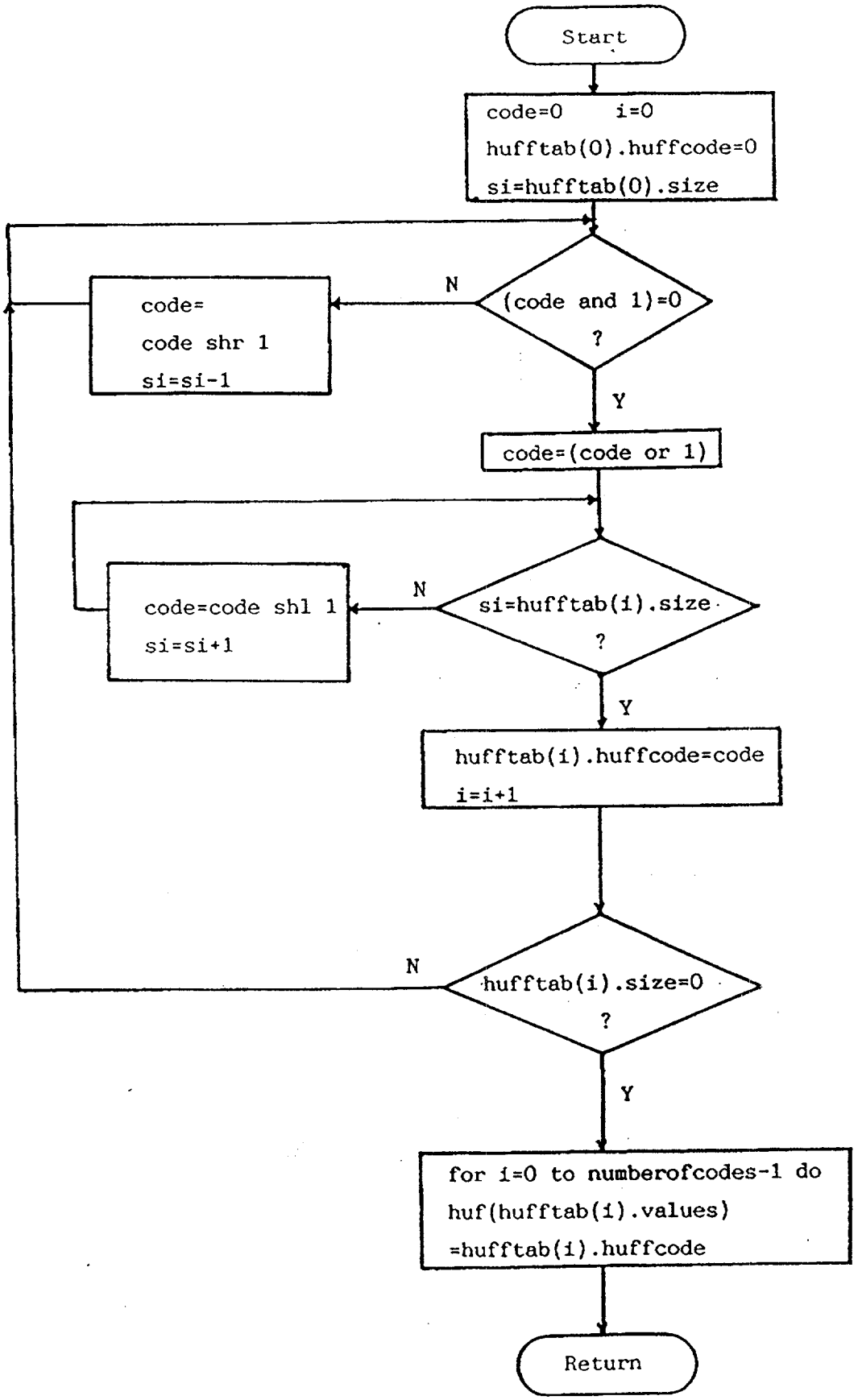
Flow 4.5.2.3 **Make the codes**



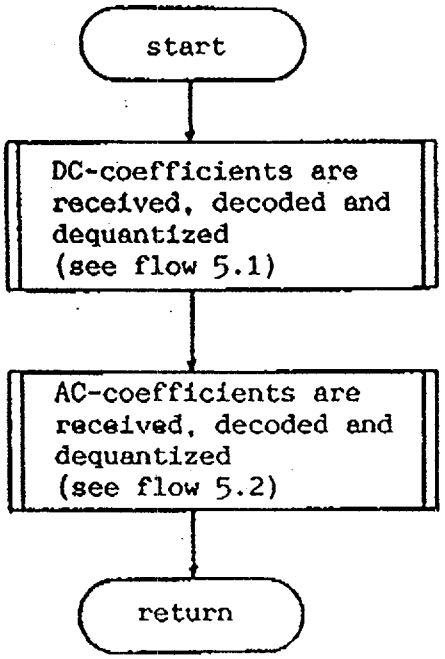
Flow 4.5.2.3.1 **Make table of codesize**



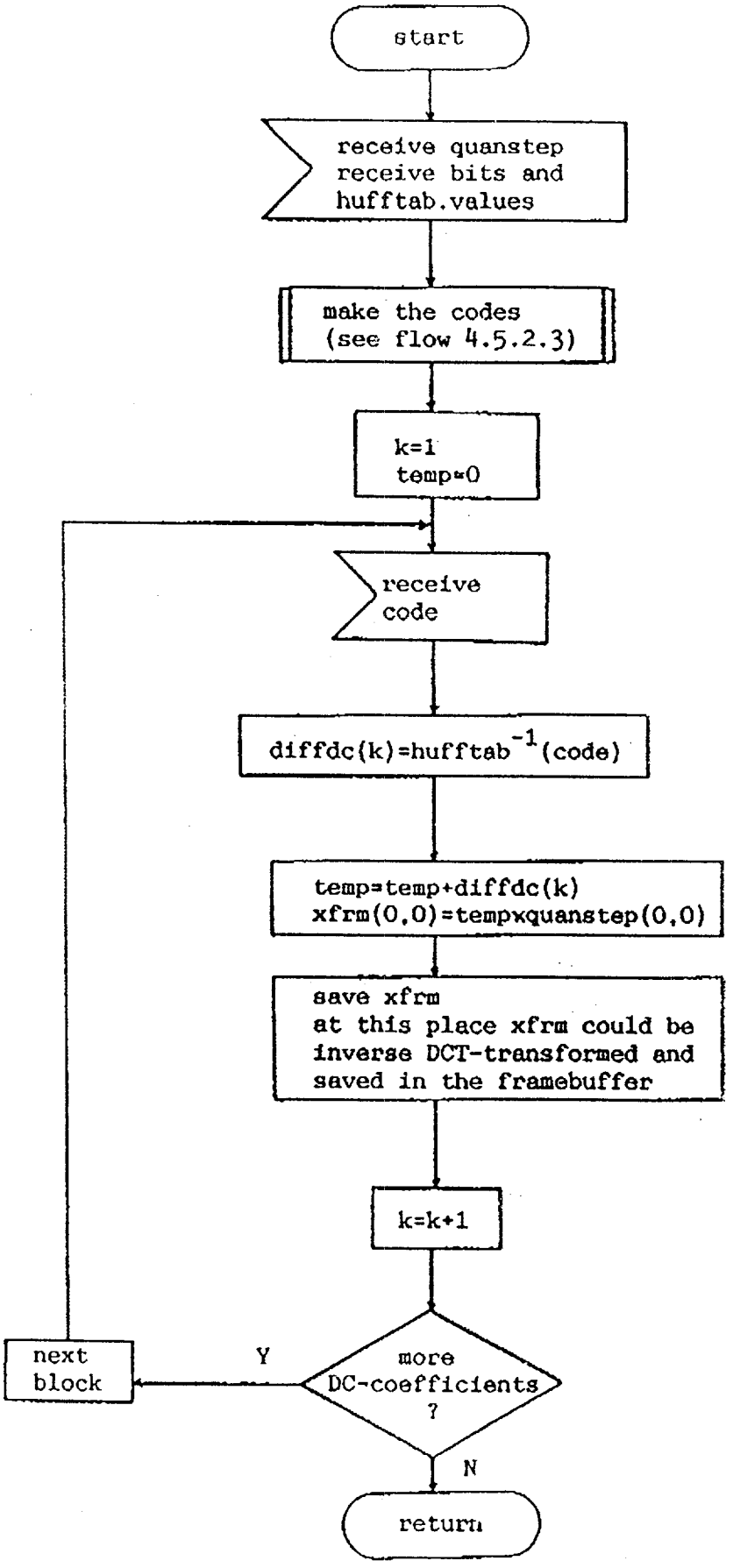
Flow 4.5.2.3.2 **Build code-table**



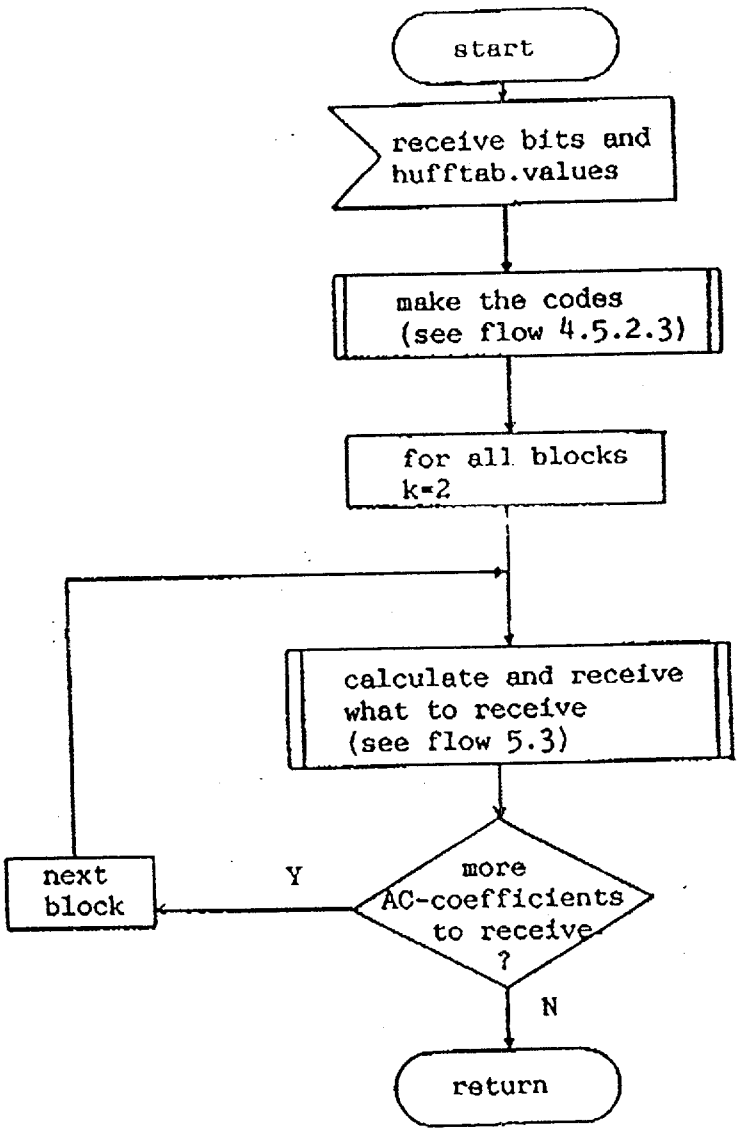
Flow 5 Receiver, Decoder and Dequantizer



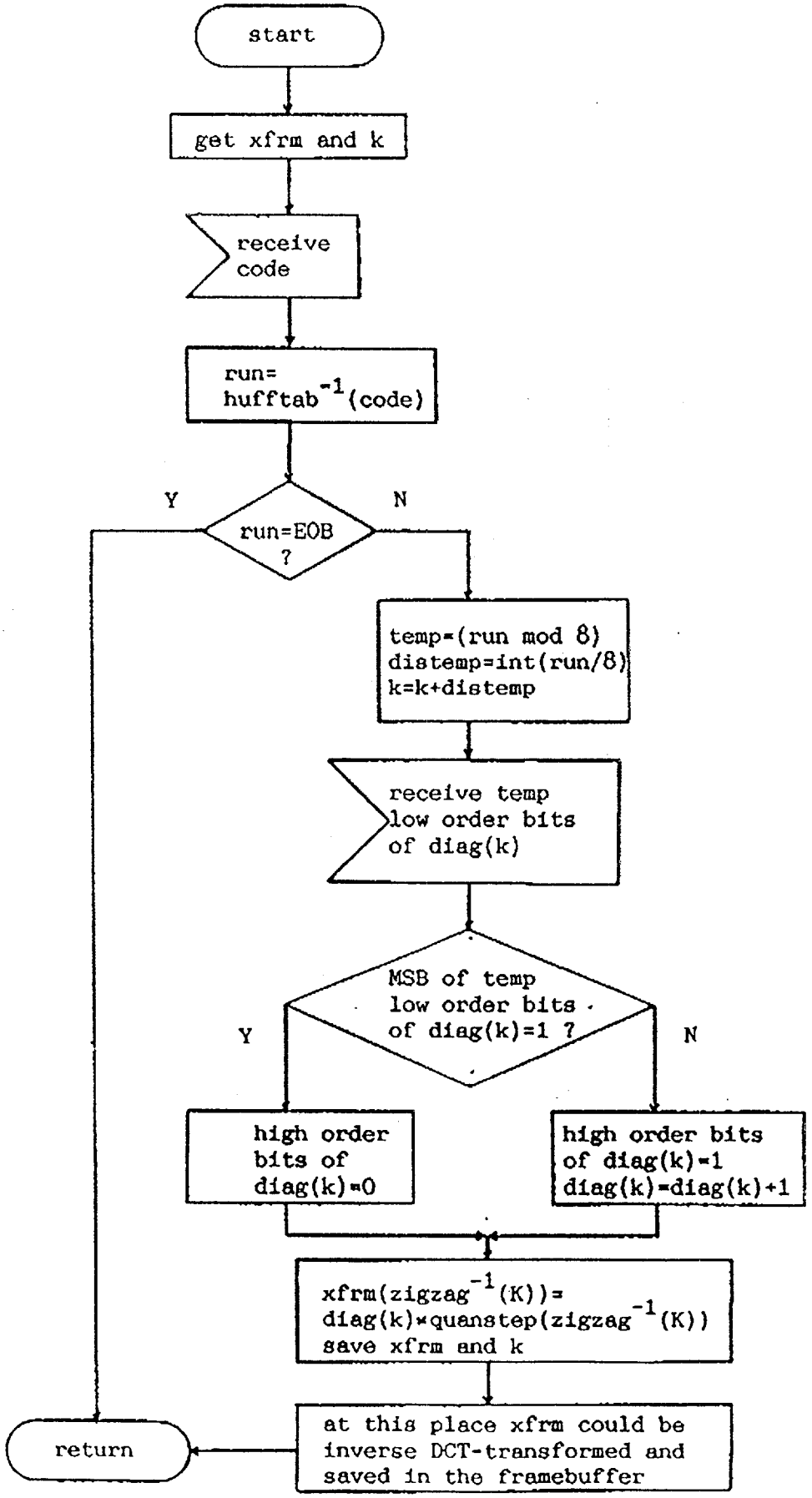
Flow 5.1 DC-coefficients are received, decoded and dequantized



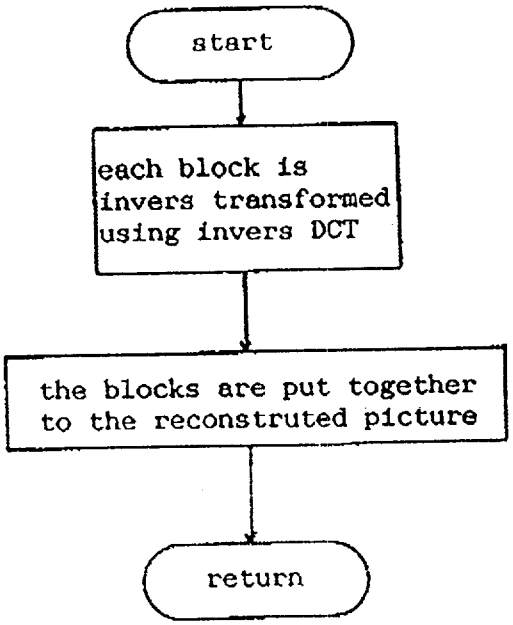
Flow 5.2 AC-coefficients are received, decoded and dequantized



Flow 5.3 Calculate and receive what to receive

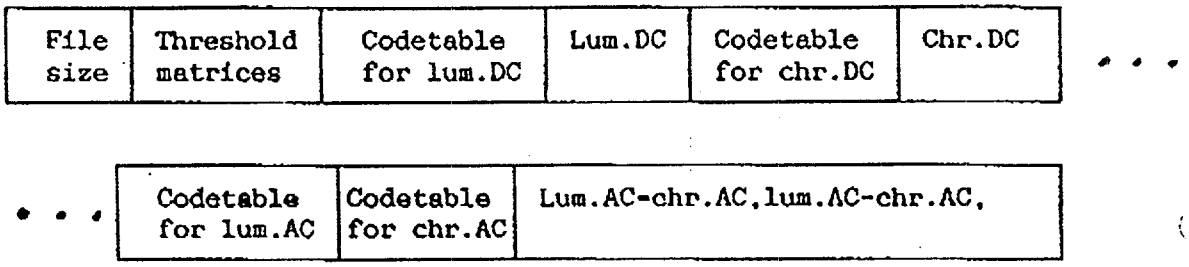


Flow 6 Inverse transform

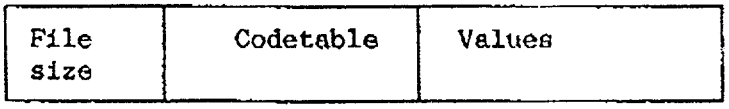


2.2.2 Syntax

Syntax for a main stage (non reversible)



Syntax for the reversible stage



2.3 SEQUENTIAL CODING

2.3.1 Flow Charts and Explanations

To be delivered later as agreed in n599 rev.1

2.3.2 Syntax

To be delivered later as agreed in n599 rev.1

2.4 FUNCTIONAL RELATIONS BETWEEN PROGRESSIVE AND SEQUENTIAL

To be delivered later as agreed in n599 rev.1

Annex A. Tables

```
(( 16, 11, 10, 16, 24, 40, 51, 61),
 ( 12, 12, 14, 19, 26, 58, 60, 55),
 ( 14, 13, 16, 24, 40, 57, 69, 56),
 ( 14, 17, 22, 29, 51, 87, 80, 62),
 ( 18, 22, 37, 56, 68,109,103, 77),
 ( 24, 35, 55, 64, 81,104,113, 92),
 ( 49, 64, 78, 87,103,121,120,101),
 ( 72, 92, 95, 98,112,100,103, 99))
```

Table 1 Visibility tresholds for the luminance.

```
(( 17, 18, 24, 47, 66, 99, 99, 99),
 ( 18, 21, 26, 66, 99, 99, 99, 99),
 ( 24, 26, 56, 99, 99, 99, 99, 99),
 ( 47, 66, 99, 99, 99, 99, 99, 99),
 ( 99, 99, 99, 99, 99, 99, 99, 99),
 ( 99, 99, 99, 99, 99, 99, 99, 99),
 ( 99, 99, 99, 99, 99, 99, 99, 99),
 ( 99, 99, 99, 99, 99, 99, 99, 99)),
```

Table 2 Visibility tresholds for crominance.

```
 1  2  6  7 15 16 28 29
 3  5  8 14 17 27 30 43
 4  9 13 18 26 31 42 44
10 12 19 25 32 41 45 54
11 20 24 33 40 46 53 55
21 23 34 39 47 52 56 61
22 35 38 48 51 57 60 62
36 37 49 50 58 59 63 64
```

Table 3 Showing how the block may be zig zag scanned

Stage number :	Description of how the picture to be encoded is calculated :
1	The original picture is filtered using the filter described in table 6. The filtered picture is subsampled 2:1 in both x- and y-direction and saved in TEMP picture. The subsampled picture is filtered and subsampled using the same procedure. The resulting picture is coded to 0.08 bit/pix using the N502 in flow 1.
2	The picture produced in stage 1. in table 5. is subtracted from the TEMP picture. The resulting picture is coded to 0.17 bit/pix using the N502 in flow 1.
3	The picture produced in stage 2. in table 5. is subtracted from the original. The resulting picture is coded to 0.5 bit/pix using the N502 in flow 1.
4	The picture produced in stage 3. in table 5. is subtracted from the original. The resulting picture is coded to 1.5 bit/pix using the N502 in flow 1.
5	The picture produced in stage 4. in table 5. is subtracted from the original. The resulting picture is huffman-coded and transmitted.

Table 4 Describing how the picture to be encoded is calculated in the different stages.

Stage number :	Description of how the reconstructed picture is updated :
1	The picture is expanded 1:2 in both x- and y-direction using the interpolation described in table 7. (Only at the decoder, the picture is expanded again to full size.)
2	The picture is added to the picture produced in stage 1. The resulting picture is expanded 1:2 in both x- and y-direction using an interpolation described in table 7.
3	The picture is added to the picture produced in stage 2.
4	The picture is added to the picture produced in stage 3.
5	The picture is received and huffman-decoded, and added to the picture produced in stage 4.

Table 5 Describing how the reconstructed picture is updated.

1	4	1
4	16	4
1	4	1

Table 6 A low pass filter.

a	x2	b
x5	x1	x3
d	x4	c

Notation:

a,b,c,d :pixel values picked in the subsampled picture

x1,x2,x3,x4,x5 :pixels values calculated as follows:

x1=(a+b+c+d)/4
x2=(a+b)/2
x3=(b+c)/2
x4=(c+d)/2
x5=(d+a)/2

Table 7 Describing how the picture is expanded.

Annex B. Alternative for reversibility

DESCRIPTION OF THE METHOD USED TO GET THE REVERSIBILITY

We tried several possibilities in the pel domain (VLC of the differences, prediction of the differences, VLC of the configurations of the differences in a 2 x 2 sub-blocks) without reaching good results.

The method giving the best performances is based on two steps:
- transmission with VLCs of the remainders of the transform coefficients
- transmission of the small residual difference between original and decoded image using run-length coding of the zeroes' values and VLC of the differences.

The assumptions made for the first step are the following:
1. the transform coefficients are represented with 12 bits
2. at the previous step (2.25 bit/pel) a maximum step size of 8 is used (according the quantisation rule of fig. 1), resulting in a residual difference in the range from -3 to +4
3. the VLC used to code the Y coefficient differences is shown in tab. I
4. the VLC used to code the coefficient differences of the chrominance components (CR and CB) is shown in tab. II

The assumption made for the second step are:
1. the VLC used to code the zeroes sequences is written in tab. III.
2. the last word (100110) is used both to say that no errors were encountred in the 90 pels examined and to form an address greater than 720 which serves to pass to the following line
3. the VLC used to code the pel-value differences is shown in tab. IV.

The statistics used to compute the different VLCs can of course be improved but, we aspect, with marginal effect on the whole performances.

-3	0.1731457412	3	000
-2	0.2661795914	2	01
-1	0.1864149272	2	11
0	0.1227117106	3	100
1	0.0886115953	4	0010
2	0.0666907802	4	0011
3	0.0525824651	4	1010
4	0.0436631963	4	1011

E = 2.80 bit

Table 1

-3	0.2682926953	2	01
-2	0.3902438879	1	1
-1	0.1731707305	3	000
0	0.0731707290	4	0011
1	0.0414634161	5	00101
2	0.0243902430	6	001001
3	0.0170731712	7	0010000
4	0.0121951215	7	0010001

E = 2.29 bit

Table 2

-2	0.0000149560	3	111
-1	0.3846521974	2	10
1	0.6152132154	1	0
2	0.0001196476	3	110

Table 3

0	0.0947560295	3	110
1	0.0906299502	4	0000
2	0.0784915835	4	0010
3	0.0741735846	4	0100
4	0.0661373138	4	0110
5	0.0599241965	4	0111
6	0.0511442684	4	1010
7	0.0487453826	4	1011
8	0.0427001864	5	00010
9	0.0394377001	5	00011
10	0.0351436920	5	01010
11	0.0307777189	5	10000
12	0.0282828771	5	10010
13	0.0241807792	5	11101
14	0.0220937487	5	11111
15	0.0191431176	6	001101
16	0.0182795189	6	001111
17	0.0170320980	6	010110
18	0.0140814660	6	100011
19	0.0133618005	6	100111
20	0.0113707241	6	111100
21	0.0113227461	6	111101
22	0.0096915029	7	0011001
23	0.0088758813	7	0011101
24	0.0079403156	7	0101111
25	0.0063330615	7	1110001
26	0.0065489612	7	1110000
27	0.0055174399	7	1110011
28	0.0046778293	8	01011100
29	0.0048217629	8	00111001
30	0.0038382190	8	10001001
31	0.0039821523	8	10001000
32	0.0034304082	8	10001011
33	0.0030465864	8	11100101
34	0.0027107422	9	001100001
35	0.0024708535	9	001110000
36	0.0024948423	9	001100011
37	0.0019191095	9	010111011
38	0.0017751763	9	100010101
39	0.0013913544	10	0011000000
40	0.0012234324	10	0011000101
41	0.0010075325	10	1000101000
42	0.0011754546	10	0011100011
43	0.0013913544	10	0011000001
44	0.0010555102	10	0101110101
45	0.0006956772	11	00110001000
46	0.0005757328	11	00111000101
47	0.0007676438	10	1110010001
48	0.0004317996	11	10001010011
49	0.0005757328	11	01011101000
50	0.0006237106	11	00111000100
51	0.0003838219	11	11100100001
52	0.0004557885	11	10001010010
53	0.0002638776	12	010111010010
54	0.0003358442	12	001100010010
55	0.0001919109	12	111001000001
56	0.0002398887	12	010111010011
57	0.0001919109	12	111001001100
58	0.0001439332	13	0011000100111
59	0.0001679221	13	0011000100110

60	0.0001919109	12	111001001101
61	0.0001919109	12	111001001010
62	0.0001919109	12	111001001011
63	0.0000959555	13	1110010000001
64	0.0000479777	14	11100100000001
65	0.0001919109	12	111001001000
66	0.0000479777	14	11100100111100
67	0.0000479777	14	11100100111101
68	0.0000239889	15	111001001111100
69	0.0000479777	14	11100100100110
70	0.0000479777	14	11100100100111
71	0.0000239889	15	111001001111101
72	0.0000719666	14	11100100000000
73	0.0000479777	14	11100100100100
74	0.0000479777	14	11100100100101
75	0.0000239889	15	111001001110010
76	0.0000479777	14	11100100111010
77	0.0000479777	14	11100100111011
78	0.0000000001	18	111001001111111010
79	0.0000239889	15	111001001110011
80	0.0000239889	15	111001001110000
81	0.0000000001	18	111001001111111011
82	0.0000000001	18	111001001111111000
83	0.0000000001	18	111001001111111001
84	0.0000239889	15	111001001110001
85	0.0000000001	18	111001001111111110
86	0.0000000001	18	111001001111111111
87	0.0000000001	18	111001001111111100
88	0.0000239889	15	111001001111110
89	0.0000000001	18	111001001111111101
90	0.0138895549	6	100110

Table 4

Annex C. Results with predefined VLCs

ISO

INTERNATIONAL ORGANISATION FOR STANDARDIZATION
ORGANISATION INTERNATIONALE DE NORMALISATION

ISO/TC97/SC2/WG8

CODED REPRESENTATION OF PICTURE
AND AUDIO INFORMATION

ISO/TC97/SC2/WG8 N

/adctg N 39

Source: SAT/CCETT

Title : Symetry for the ADCT scheme - N 502

Introduction

As agreed at our Washington meeting, CCETT has continued the work for the full symetry of N 502. We have always STRESSED the need for this specific item, for at least the following reasons :

1. Simplicity of implementation e.g.on silicon
2. ONE - PASS Coding technique
3. Greater Compatibility with Visiophony, Teleconference and Digital T.V. Coding techniques
4. Better fitness to Real Time constraints, and especially for Face to Face services

Results

The following results are based on either the use of Customized VLC as described in N 502, or the use of Predefined VLC tables derived from the four ESPRIT test images. In most cases the impairments on the coding rate are negligible for "natural images", and very limited on "binary images" - note that the learning phase does not include "binary images", yet.

Pictures	Coding rate n502	Coding rate with predefined VLCs
Barbara	1.03	1.02
Binary1	1.35	1.38
Binary2	0.59	0.63
Binary3	0.66	0.70
Binary4	0.97	1.01
Zelda	0.55	0.56
Clown	0.86	0.86
Flower	0.85	0.85
Boy	0.99	0.99
Iba	0.92	0.92
Toys	0.79	0.79
EbuChar	0.73	0.74
Boats	0.92	0.92
Tree	1.51	1.52
Violin	1.56	1.58
Young	0.90	0.91

Annex D. Overall Results

List of the illustrations