MPEG-2 over ATM

John Koiste

Senior Engineer

Digital video and digital multimedia compression techniques, such as MPEG-2, are among the most exciting and important developments in electronics and communications in the 1990s. MPEG-2 video and audio, in one form or another, are likely to become major traffic generators for ATM networks sometime during the next few years. This session first offers a fairly detailed description of MPEG-2 and then describes NetCalc3—a multimedia network design program for "production" network design—which, among other things, models the Transport Stream (TS), the systems layer of the MPEG-2 compressed video/audio/data multiplexing and packetization technique.

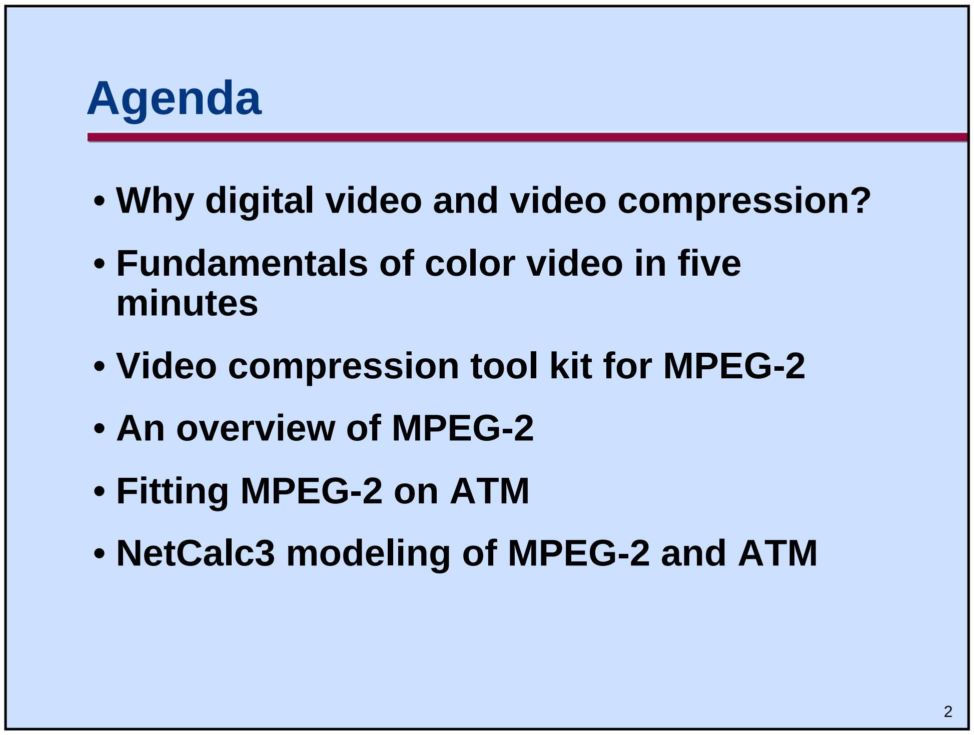
The objective of this workshop is to share MPEG-2 and ATM network design information with Nortel customers. Nortel’s network engineering experience indicates that there is a strong correlation between understanding of applications, such as MPEG-2 is to ATM, and quality and speed of network design.

About the presenter:

John Koiste has been associated with the Magellan product line for more than 14 years, first in R&D at Nortel Technolgy (formerly Bell-Northern Research), then in product management, marketing, technical consulting, competitive analysis, and network engineering at Nortel.

The last four years at Nortel have been spent working on network designs with Nortel engineers and customers around the world, and designing computer-aided tools to make Magellan network engineering easier, faster and more accurate.

Prior experience with three other companies includes the design of aircraft radar, navigation computers, local area networks, special-purpose small computers, and CAD/CAM. Koiste was a guest lecturer at McGill University in Montreal for several years, teaching second-level digital and analog electronics courses to electrical engineering undergraduates.



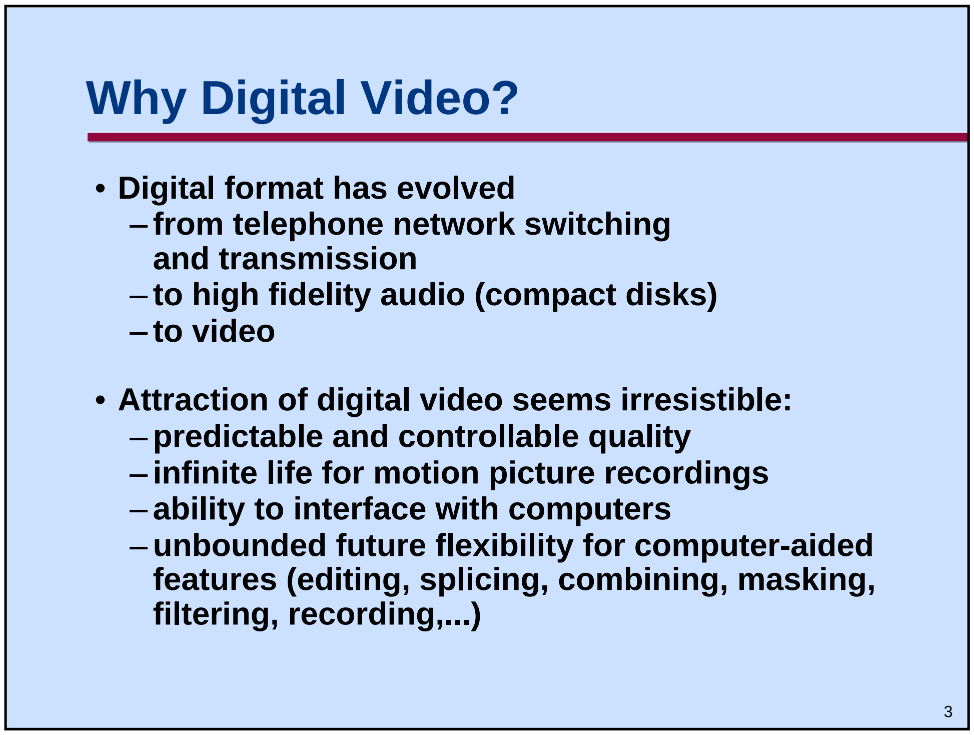
Agenda

- Why digital video and video compression?
- Fundamentals of color video in five minutes
- Video compression tool kit for MPEG-2
- An overview of MPEG-2
- Fitting MPEG-2 on ATM
- NetCalc3 modeling of MPEG-2 and ATM

2

In order to make this presentation more self-contained, it includes a preamble on why digital video and video compression are important. There is also a brief overview on the fundamentals of video itself because it is necessary to be familiar with common video terminology before embarking on video compression methods.

This presentation provides enough information about video for network designers to be able to design networks that need to carry MPEG-2 traffic. The references at the end can be used for further learning about the subject.



Why Digital Video?

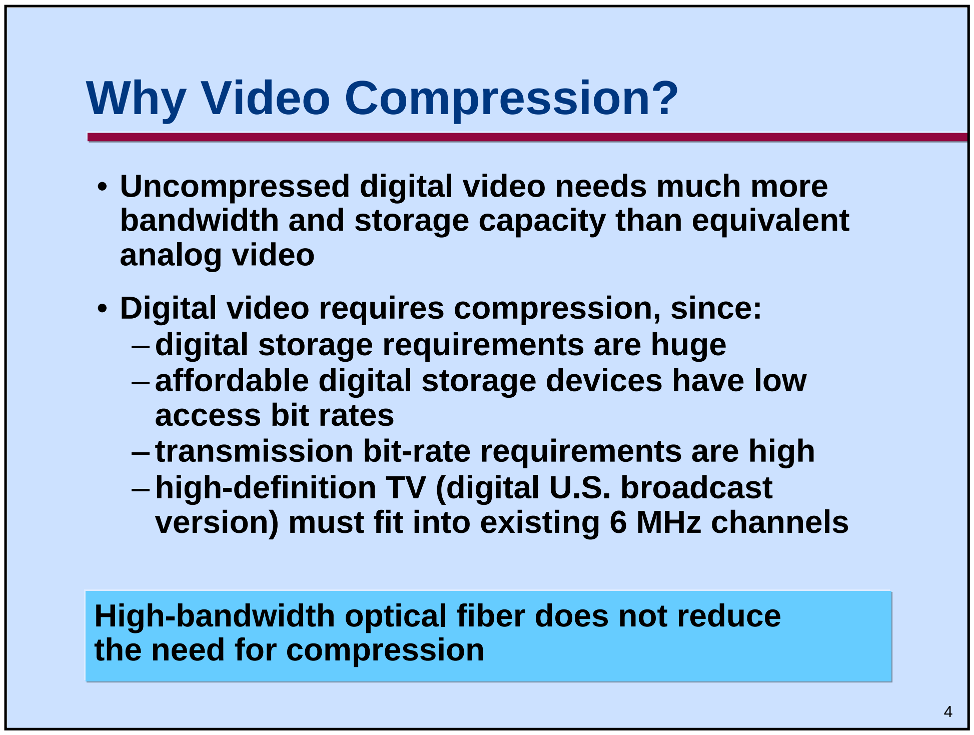
- Digital format has evolved
  - from telephone network switching and transmission
  - to high fidelity audio (compact disks)
  - to video
- Attraction of digital video seems irresistible:
  - predictable and controllable quality
  - infinite life for motion picture recordings
  - ability to interface with computers
  - unbounded future flexibility for computer-aided features (editing, splicing, combining, masking, filtering, recording,...)

3

Implementing digital video requires more advanced circuits than for digitizing of the telephone system or for digital high fidelity audio because of the enormous amount of associated information and high resulting data rate.

But the drive to digital format is compelling and has already proven to be tremendously successful in digital satellite television broadcasting in North America and Europe. Much of digital video distribution will involve satellite communication methods instead of terrestrial fiber, but there is so much untapped potential for application of digital video that terrestrial networks will see their full share of video in the future. For applications requiring fast interaction, as in video conferencing, terrestrial networks offer acceptable delays while satellite systems are only marginally acceptable because of propagation delay of signals.

Over the next few years, the digital format will totally revolutionize the creation of motion pictures, business videos and home videos because it brings computers into the process in an elegant and effective way.



Why Video Compression?

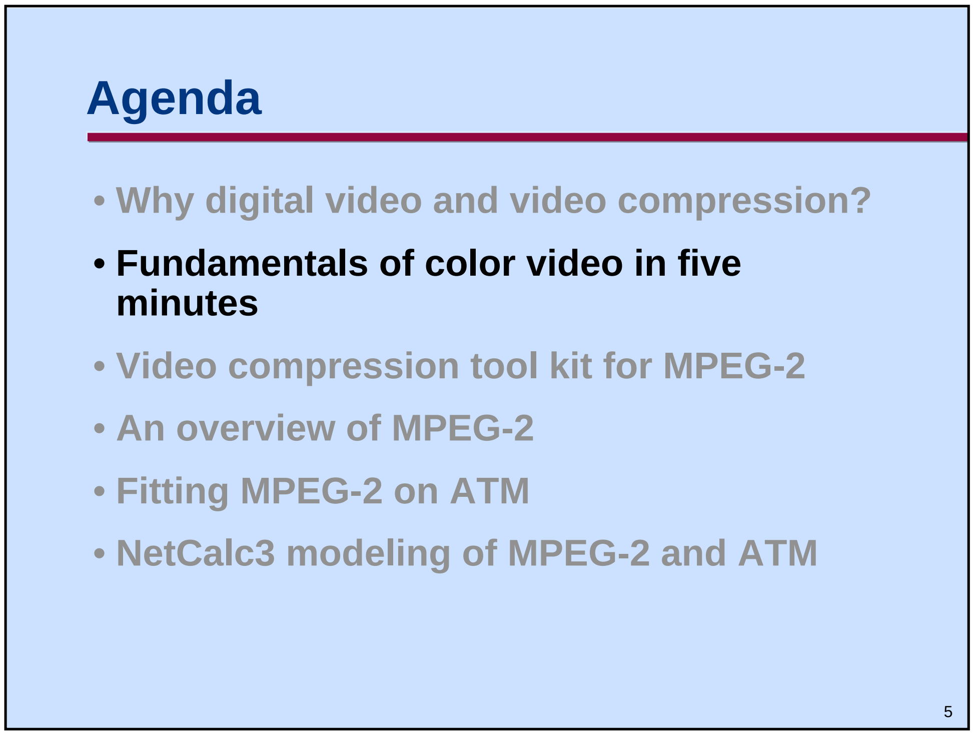
- Uncompressed digital video needs much more bandwidth and storage capacity than equivalent analog video
- Digital video requires compression, since:
  - digital storage requirements are huge
  - affordable digital storage devices have low access bit rates
  - transmission bit-rate requirements are high
  - high-definition TV (digital U.S. broadcast version) must fit into existing 6 MHz channels

High-bandwidth optical fiber does not reduce the need for compression

4

The downside of the digital format, and the reason it took so long to appear, is that in its raw (uncompressed) format, it requires huge amounts of transmission bandwidth and memory capacity for storage compared to analog video of the same quality. (Examples of how much of each are given later.) The need to go digital had to wait for technology to catch up. Perhaps the final turning point was the clear superiority of digital high definition television performance against analog HDTV in 1991-92 tests done in the U.S. In those tests, the compression techniques (now used in MPEG-2) were proven to be feasible beyond any doubt—and to offer better performance in all key categories than competing analog systems. One might ask why the U.S. HDTV requirement above is listed with the first three generic requirements for compression. The reason is that there was a major spinoff of technology because of the U.S. HDTV effort (which also involved several non-U.S. companies) into MPEG-2 and into European HDTV. MPEG-2, as noted later, has great commonality with digital HDTV compression and coding methods.

Increasing transmission bandwidth, for example with fiber, does not reduce the need for video compression. The need is there for the other three reasons above, and its fulfillment also happens to reduce wide-area bandwidth requirements. But even with compression, video will still need a lot of bandwidth because its application potential is so great—and that potential is all the greater because of the viability of compression.

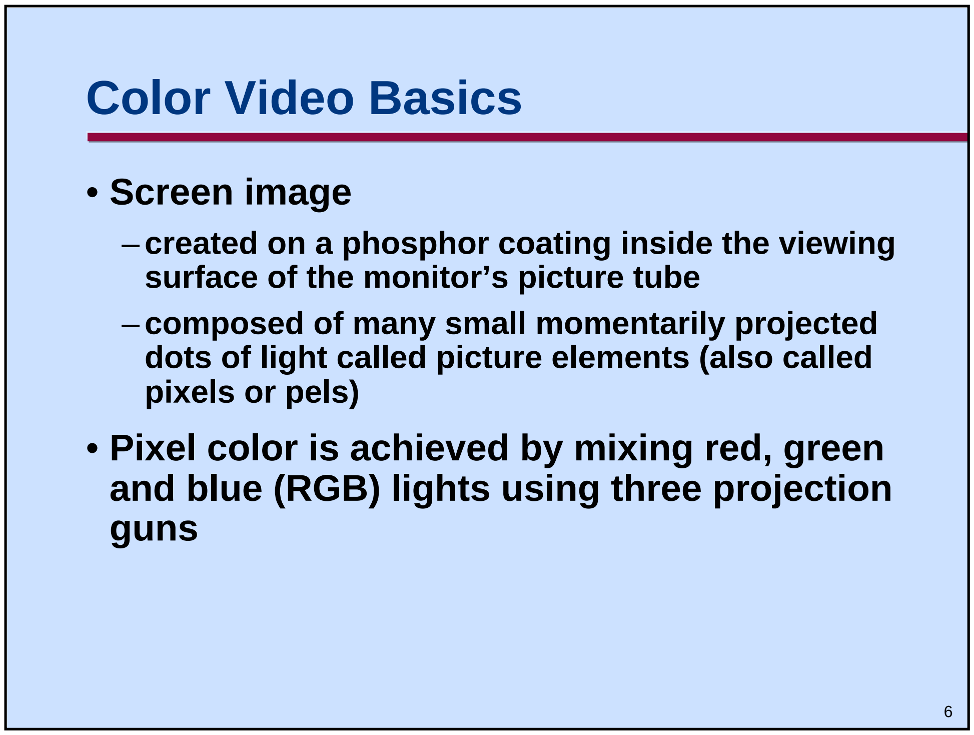


Agenda

- Why digital video and video compression?
- **Fundamentals of color video in five minutes**
- Video compression tool kit for MPEG-2
- An overview of MPEG-2
- Fitting MPEG-2 on ATM
- NetCalc3 modeling of MPEG-2 and ATM

5

The following several slides are modified versions from Inform ‘95 and contain material familiar to most readers. They have been included to make the presentation more self-contained.

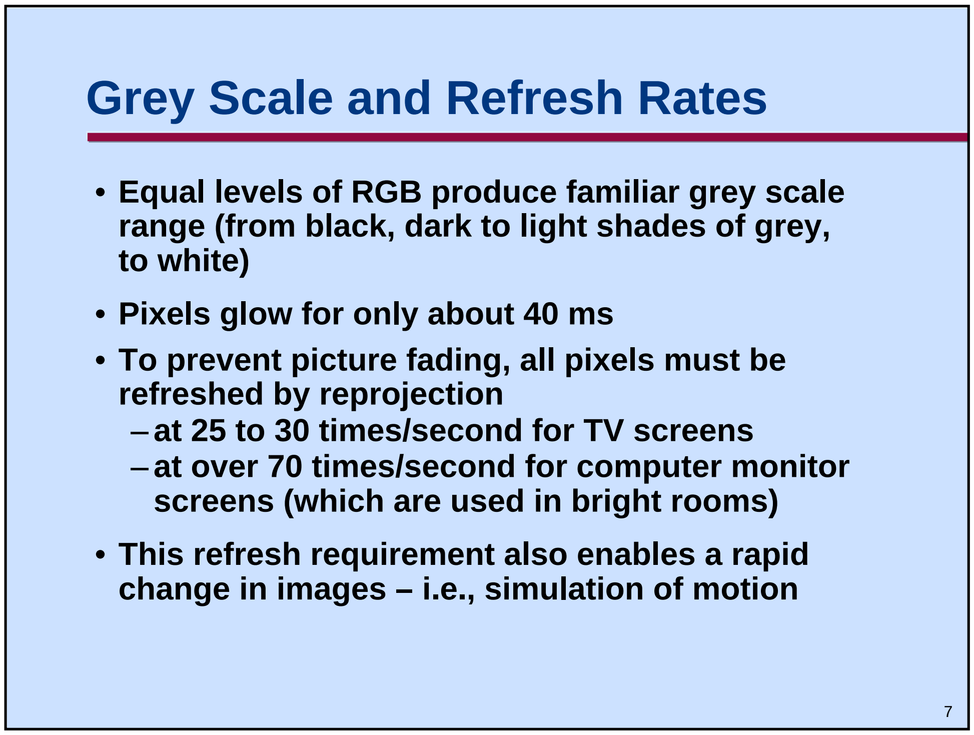


Color Video Basics

- Screen image
  - created on a phosphor coating inside the viewing surface of the monitor’s picture tube
  - composed of many small momentarily projected dots of light called picture elements (also called pixels or pels)
- Pixel color is achieved by mixing red, green and blue (RGB) lights using three projection guns

6

Color on a video screen uses the additive red-green-blue (RGB) color mixing technique while printing color or mixing water colors is a subtractive technique that uses cyan (often wrongly called “blue”), magenta (wrongly called “red”) and yellow (collectively called CMY).



Grey Scale and Refresh Rates

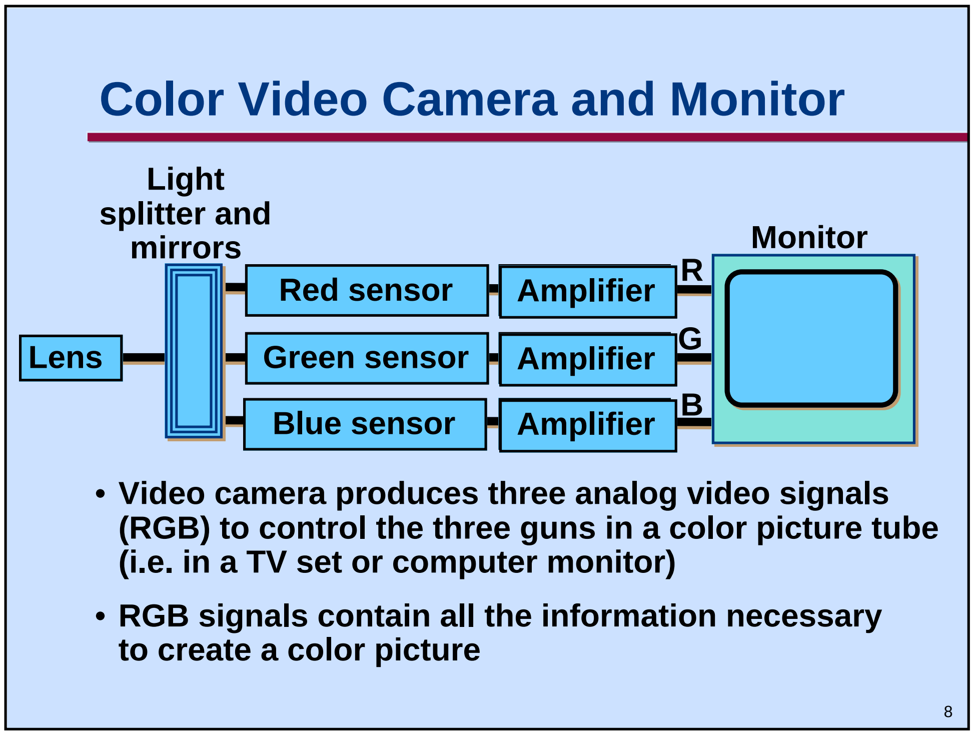
- Equal levels of RGB produce familiar grey scale range (from black, dark to light shades of grey, to white)
- Pixels glow for only about 40 ms
- To prevent picture fading, all pixels must be refreshed by reprojection
  - at 25 to 30 times/second for TV screens
  - at over 70 times/second for computer monitor screens (which are used in bright rooms)
- This refresh requirement also enables a rapid change in images – i.e., simulation of motion

7

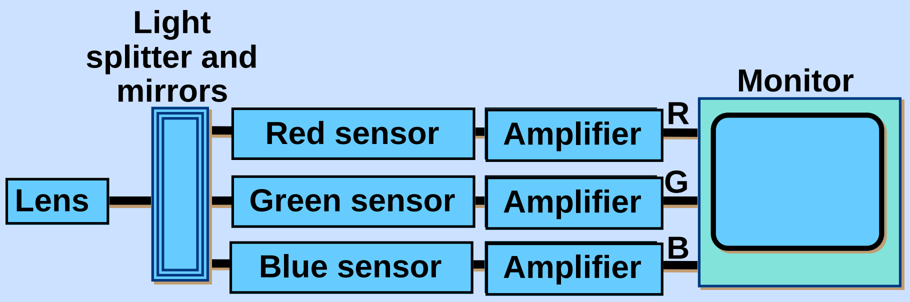
With the additive technique, equal intensities of RGB always produces a monochrome light. The luminance of the white light so produced is a function of the intensity level of the three additive primaries. Black is achieved with zero intensity of RGB. If the intensities of RGB are kept equal and their intensity starts at zero and is then gradually increased to maximum, the resulting color starts at black, changes to a dark grey, then to lighter shades of grey and finally to white. This is called the grey scale in monitors.

Pixel life (glow time) is a function of several parameters, including the persistence of the phosphor coating on the inside surface of the screen (or equivalent on a flat screen as on a portable computer). To allow for moving pictures, low persistence is required. Pixels must therefore be refreshed more often to retain picture brightness. Computer monitor screens are usually refreshed at 70 times per second or more since they tend to be used in a bright office environment.

With today’s regular definition television, the interlacing refresh format is used. With interlacing, odd and even lines are refreshed in an alternating manner. The group of odd lines is called the top field, while the even group is called the bottom field. It takes two field refresh cycles to refresh all pixels. Interlacing reduces transmission bandwidth (by 2:1) and also (by a very slight amount) screen brightness flicker.



Color Video Camera and Monitor

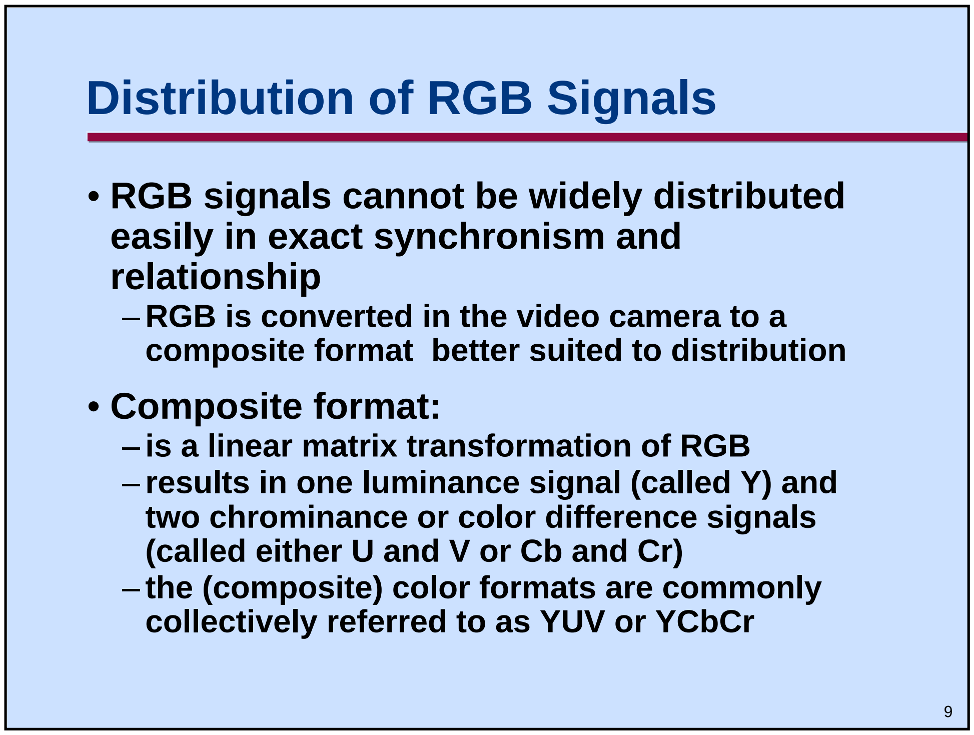


- Video camera produces three analog video signals (RGB) to control the three guns in a color picture tube (i.e. in a TV set or computer monitor)
- RGB signals contain all the information necessary to create a color picture

8

The three-sensor video camera shown for illustrative purposes above is actually an expensive type usually only seen in TV studios. Home cameras or video recorders use a much less complicated single sensor technique which produces a somewhat lower quality picture but at a much lower price.

The key point for this workshop is that RGB signals contain all the information required to produce a color picture on a TV or computer monitor screen.



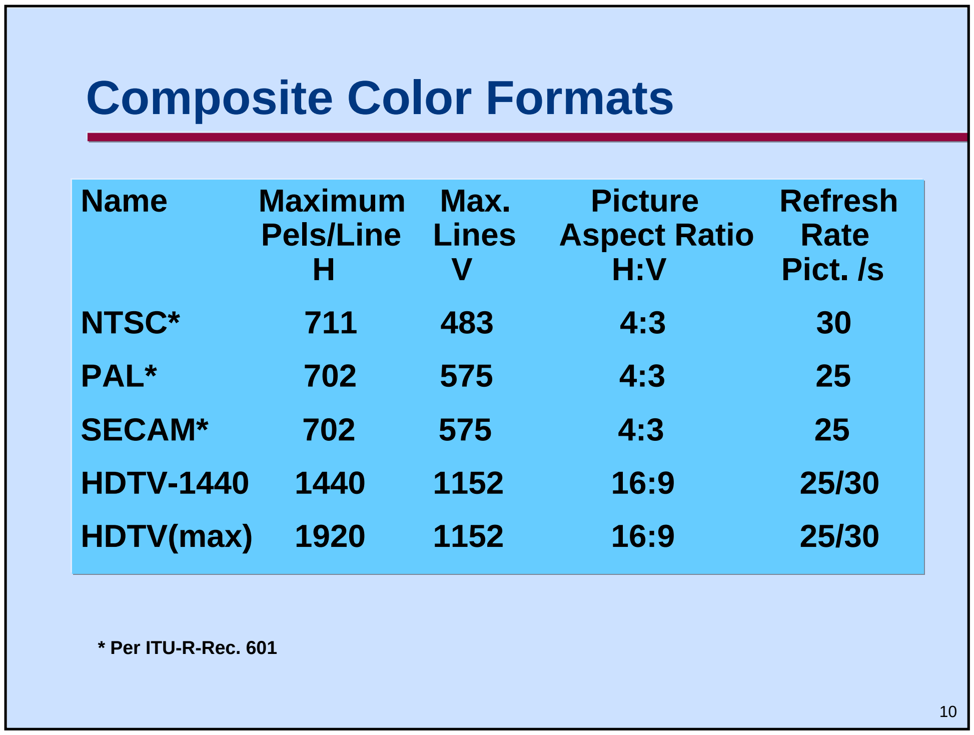
Distribution of RGB Signals

- RGB signals cannot be widely distributed easily in exact synchronism and relationship
  - RGB is converted in the video camera to a composite format better suited to distribution
- Composite format:
  - is a linear matrix transformation of RGB
  - results in one luminance signal (called Y) and two chrominance or color difference signals (called either U and V or Cb and Cr)
  - the (composite) color formats are commonly collectively referred to as YUV or YCbCr

9

Anyone who has worked in analog electronics design knows that, in general, it is very difficult to distribute multiple AC signals while keeping their levels in proportion and phases in original relationship. The RGB signals are no exception and, as a result, composite formats that are less vulnerable to such distortions were defined over 40 years ago for color television.

The composite format also allowed ongoing use of existing black-and-white television sets, at least for the NTSC variation mentioned on the next chart.



Composite Color Formats

Name	Maximum Pels/Line H	Max. Lines V	Picture Aspect Ratio H:V	Refresh Rate Pict. /s
NTSC*	711	483	4:3	30
PAL*	702	575	4:3	25
SECAM*	702	575	4:3	25
HDTV-1440	1440	1152	16:9	25/30
HDTV(max)	1920	1152	16:9	25/30

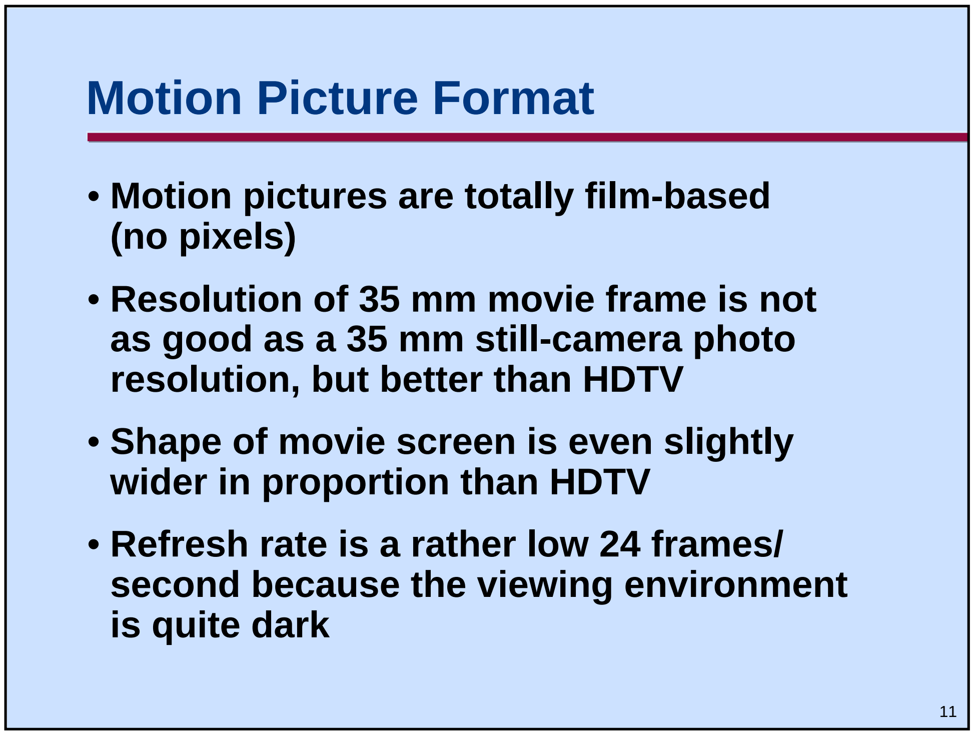
* Per ITU-R-Rec. 601

10

Time has shown that the three composite format analog systems (NTSC, PAL and SECAM) easily rank among the most important and successful applications of electronics engineering design in the last 50 years. NTSC is used in the USA, Canada, Japan and a few other countries. SECAM is used in France, Russia, all of eastern Europe, and several other countries. PAL is used in western Europe and most of the rest of the world.

The ITU-R-Rec. 601 defines the digital formats for NTSC, PAL and SECAM. The maximum visible resolutions are shown above. Note that NTSC has a nominal 525 lines, but only a maximum of 483 pixels are visible in the vertical direction, and that is often reduced to 480. In fact, the resolution often used is 640x480 instead of the maximum 711x483 shown above. Similarly, PAL and SECAM have 625 lines but a maximum of 575 are visible per the 601 specification. (Note that it is usual to quote resolution as HxV—not VxH—and that V is the number of visible lines when referring to television.)

HDTV has the two resolutions shown above (as maximum).



Motion Picture Format

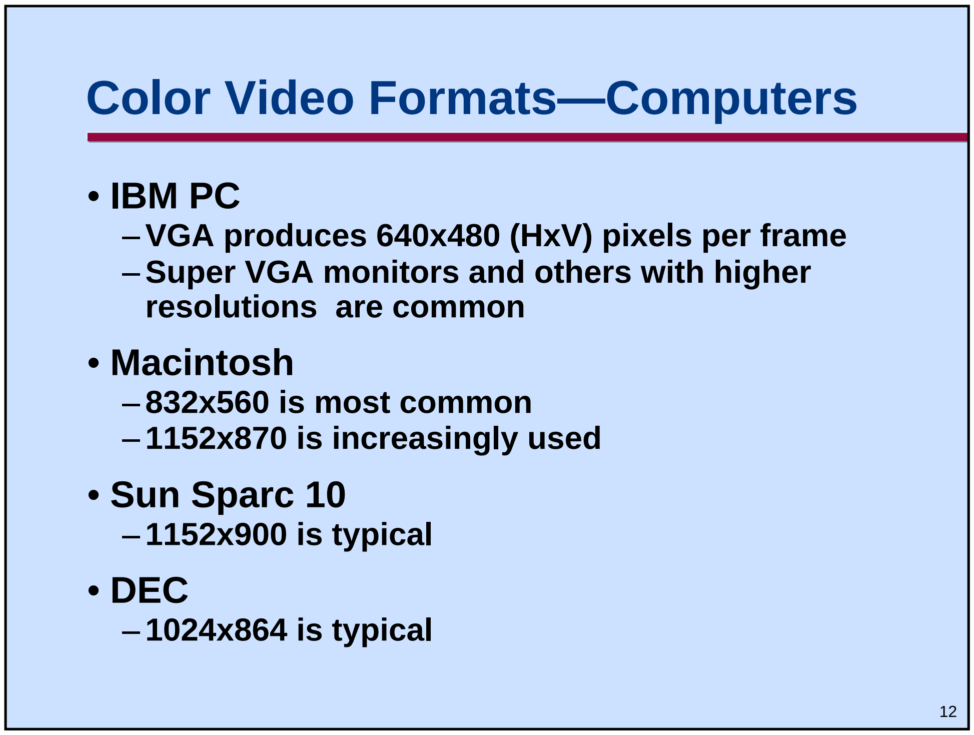
- Motion pictures are totally film-based (no pixels)
- Resolution of 35 mm movie frame is not as good as a 35 mm still-camera photo resolution, but better than HDTV
- Shape of movie screen is even slightly wider in proportion than HDTV
- Refresh rate is a rather low 24 frames/ second because the viewing environment is quite dark

11

When movies are recorded on NTSC videotape, the goal is to get an information refresh rate of 30 frames/second from 24. This is achieved by recording alternate movie frames twice and three times respectively. With this process, 12 frames every second are recorded three times each, which results in 36 fields per second, where a field is an NTSC frame (picture) with either the odd or even lines missing. The other 12 alternate frames each second are recorded twice, producing 24 fields. Therefore, 60 fields are recorded each second, and that translates to a picture rate of 30 frames per second. Obviously, successive recorded fields alternate between top or bottom field (even or odd lines missing), so that every 1/30th of a second there is a complete NTSC frame produced for the viewer. This process is called 3:2 pulldown.

As a side note, almost all video (or TV) cameras in use today are field-based (i.e. use the interlace refresh format), instead of frame- or picture-based (progressive refresh format). Using interlace instead of progressive format for television has advantages and disadvantages (see Section 5 in [7]). One of the differences between MPEG-2 and MPEG-1 is that MPEG-2 codes interlaced format more efficiently, which is very important because of the huge number of field-based video cameras in use. But the long-term trend in HDTV is toward progressive format, and MPEG-2 handles that even better.

For PAL or SECAM, where the information refresh rate is 25 frames/second, the movie is simply speeded up by 25/24, which is not objectionable, and recorded on videotape.

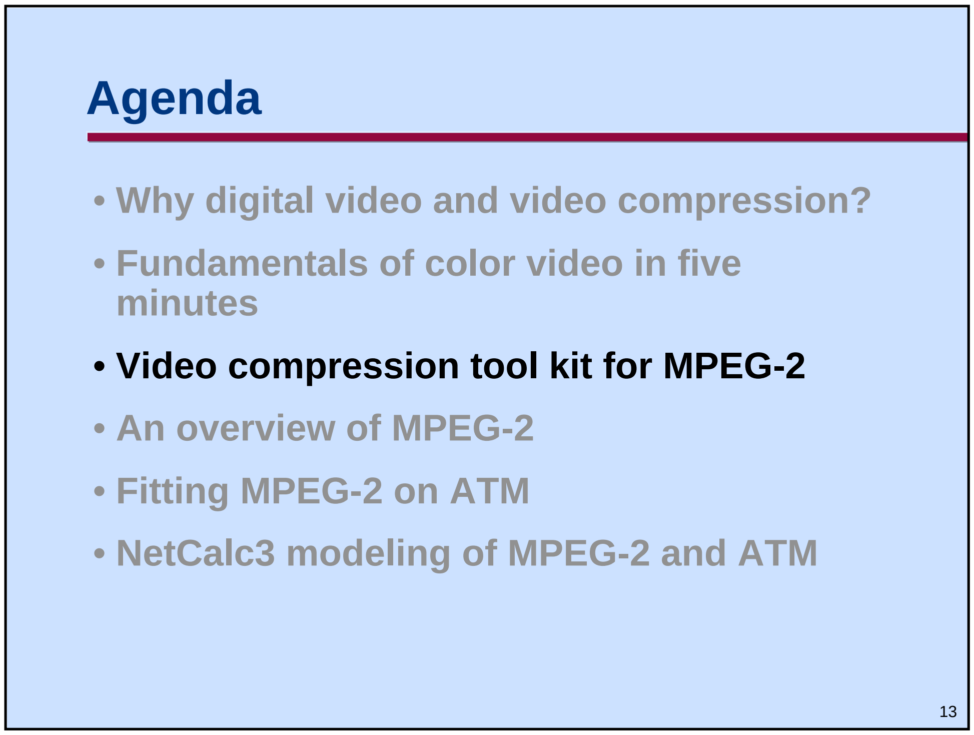


Color Video Formats—Computers

- IBM PC
  - VGA produces 640x480 (HxV) pixels per frame
  - Super VGA monitors and others with higher resolutions are common
- Macintosh
  - 832x560 is most common
  - 1152x870 is increasingly used
- Sun Sparc 10
  - 1152x900 is typical
- DEC
  - 1024x864 is typical

12

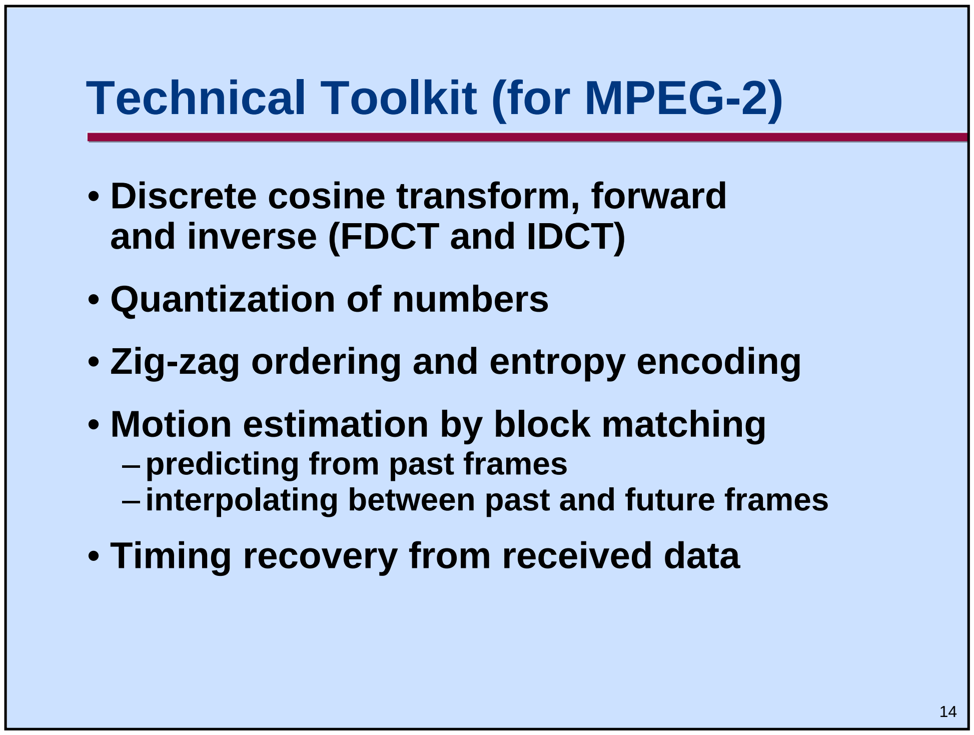
There are more screen sizes and resolutions available for popular computer types every year. The most notable aspect of the resolutions above (and others) is that the numbers are in the same ballpark but are still different—for example, compare those of the Mac and Sparc. The lack of standardization makes multimedia plug-in cards somewhat harder and more expensive to design for cross-platform applications. But MPEG-2 makes compatibility less of a problem than earlier specifications, including MPEG-1.



Agenda

- Why digital video and video compression?
- Fundamentals of color video in five minutes
- **Video compression tool kit for MPEG-2**
- An overview of MPEG-2
- Fitting MPEG-2 on ATM
- NetCalc3 modeling of MPEG-2 and ATM

13

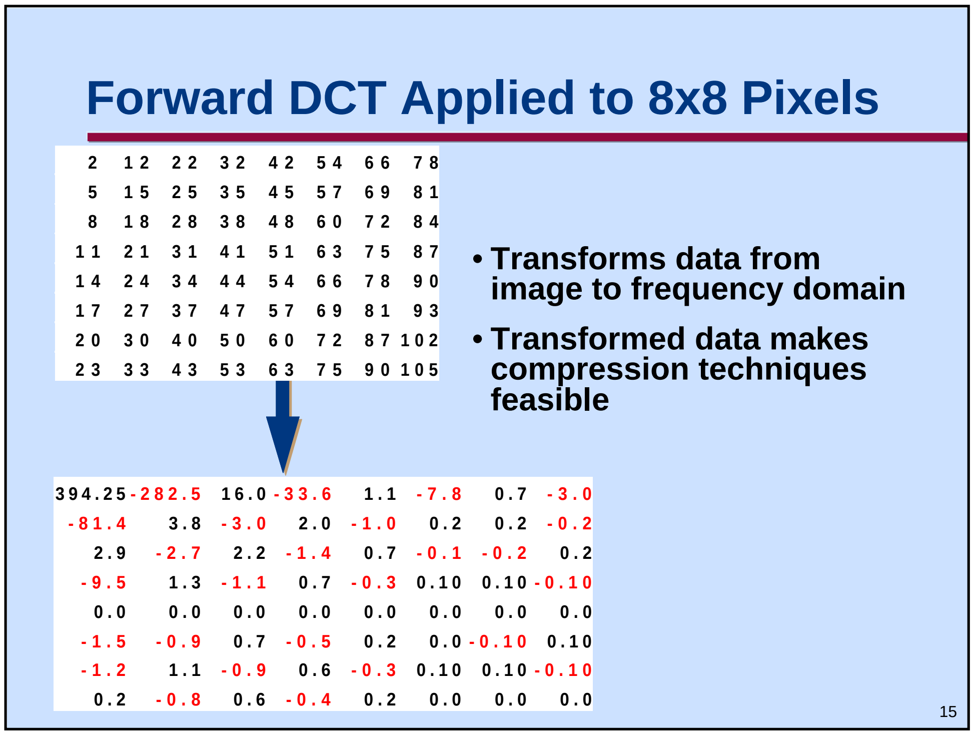


Technical Toolkit (for MPEG-2)

- Discrete cosine transform, forward and inverse (FDCT and IDCT)
- Quantization of numbers
- Zig-zag ordering and entropy encoding
- Motion estimation by block matching
  - predicting from past frames
  - interpolating between past and future frames
- Timing recovery from received data

14

The approach used in this presentation to describe MPEG-2 starts with a review of the technical methods for each key function, as listed above. Some are fairly complicated, such as DCT, while others are simple, such as quantization. But each plays an important part, and all are described briefly in the following pages. Later, in the section after this one, the key functional pieces are brought together.



Forward DCT Applied to 8x8 Pixels

2	12	22	32	42	54	66	78
5	15	25	35	45	57	69	81
8	18	28	38	48	60	72	84
11	21	31	41	51	63	75	87
14	24	34	44	54	66	78	90
17	27	37	47	57	69	81	93
20	30	40	50	60	72	87	102
23	33	43	53	63	75	90	105

- Transforms data from image to frequency domain
- Transformed data makes compression techniques feasible

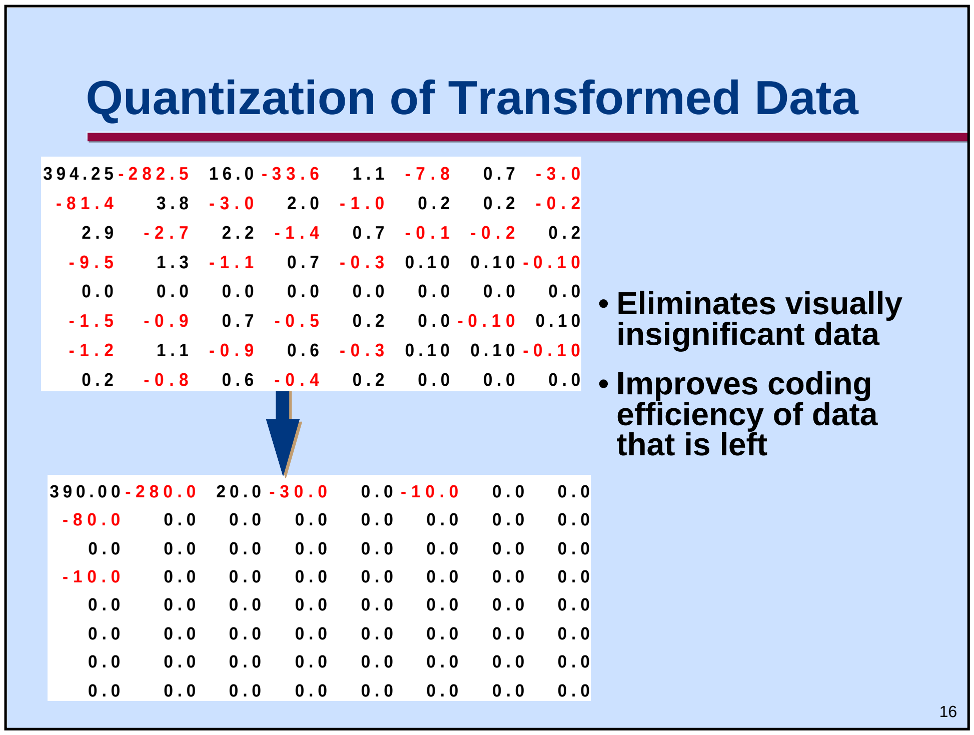
394.25	-282.5	16.0	-33.6	1.1	-7.8	0.7	-3.0
-81.4	3.8	-3.0	2.0	-1.0	0.2	0.2	-0.2
2.9	-2.7	2.2	-1.4	0.7	-0.1	-0.2	0.2
-9.5	1.3	-1.1	0.7	-0.3	0.10	0.10	-0.10
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
-1.5	-0.9	0.7	-0.5	0.2	0.0	-0.10	0.10
-1.2	1.1	-0.9	0.6	-0.3	0.10	0.10	-0.10
0.2	-0.8	0.6	-0.4	0.2	0.0	0.0	0.0

15

Transforms are used to state information in a more compact manner, by reducing redundancy, based on some statistical knowledge of that information. For example, a block of pixels 8x8 in size, randomly picked out of the picture shown, is quite likely to be of uniform color or uniform brightness because so much of the picture is composed of uniform spaces, even inside the rectangular blocks. If one could find a transform which could represent the information in an 8x8 pixel block more compactly on average—assuming the statistics as stated for the picture above—then it could open the door to efficient compression of that information. The discrete cosine transform (DCT) has proven to be the best method found to date for doing just that.

The forward and inverse DCT formulas were programmed to allow examination of any composition of an 8x8 block of pixels for encoding (compression) and decoding (attempt at recreation of the original 8x8 block). An example of such a block is shown above. The values (such as 2 in the upper left) could represent the luminance (Y) of a pixel. A separate pair of blocks would be compressed for the color difference signals (U and V).

When the 8x8 block is transformed, a new group of 64 numbers is created as shown. Note that the transformed numbers of larger size tend to be somewhat clustered in the upper left corner of the result. This is the key to compression. If the 8x8 block had any constant value, such as 75, there would be only one non-zero number in the transformed 64-number result, and it would be in the upper left corner. The DCT is a two-dimensional relative of the discrete Fourier transform, and the upper left transformed value is called the “DC” coefficient. Other terms are coefficients for “AC” terms—terms whose frequency increases to the right and down from the upper left corner of the transformed block.



Quantization of Transformed Data

394.25	-282.5	16.0	-33.6	1.1	-7.8	0.7	-3.0
-81.4	3.8	-3.0	2.0	-1.0	0.2	0.2	-0.2
2.9	-2.7	2.2	-1.4	0.7	-0.1	-0.2	0.2
-9.5	1.3	-1.1	0.7	-0.3	0.10	0.10	-0.10
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
-1.5	-0.9	0.7	-0.5	0.2	0.0	-0.10	0.10
-1.2	1.1	-0.9	0.6	-0.3	0.10	0.10	-0.10
0.2	-0.8	0.6	-0.4	0.2	0.0	0.0	0.0

390.00	-280.0	20.0	-30.0	0.0	-10.0	0.0	0.0
-80.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
-10.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

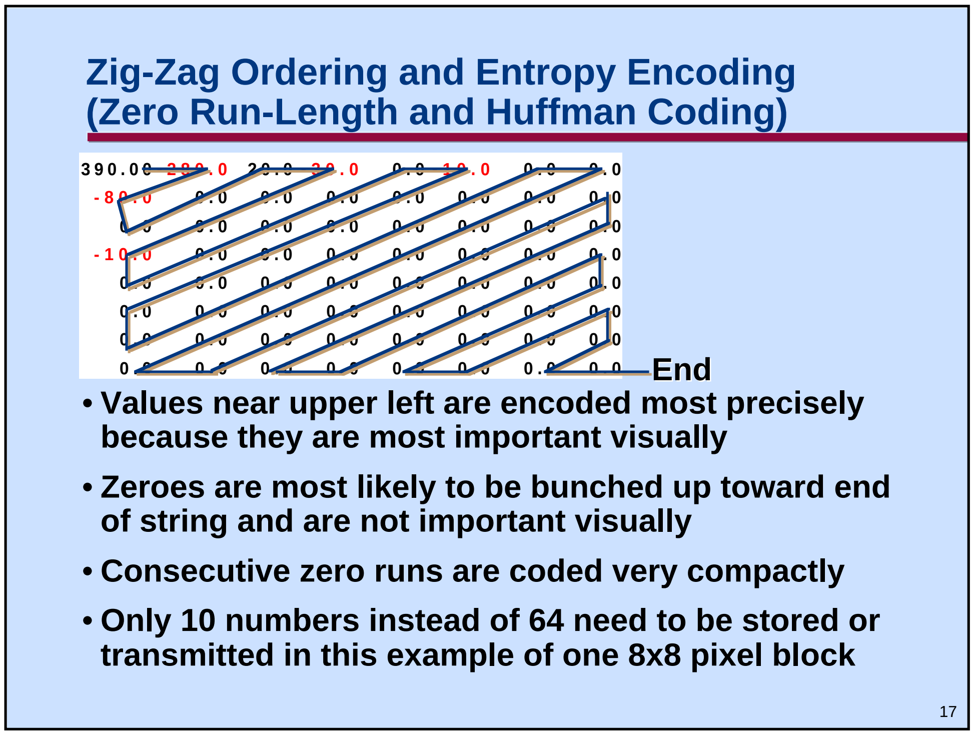
- **Eliminates visually insignificant data**
- **Improves coding efficiency of data that is left**

16

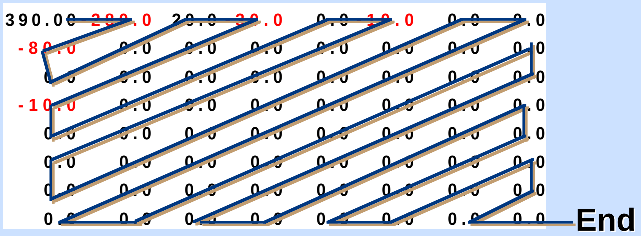
Quantization is the second step to DCT-based encoding. Most 8x8 blocks that are seen in images used by people, or in an average frame in a reel of motion picture film, will yield transformed results similar to the above. It has been observed that getting rid of values near zero, which most often occur in "higher frequency" terms (away from the upper left) does not deteriorate the decoded (recreated) image badly in a visual sense.

There are several quantization tables used in MPEG-2, and implementors of encoder products have the freedom to use any quantization method they feel is effective. The one above simply rounds the result to the nearest unit of 10 in intensity. Hence, any number between + or - 5 is set to zero, and so on. The remainder are easier to encode efficiently because they can only be a multiple of 10. The quantization makes it possible to transmit and store fewer numbers.

Is there any compression yet? No, but we're getting close, and a lot of the credit will be due to DCT and quantization. How much to each? You actually can't say that "FDCT yielded 2:1" and "quantization yielded another 3:1", and that entropy encoding (discussed next) "yielded another 3:1". You'll only be able to say that the whole process at the picture level yielded X:1, and the value of X varies with the image being compressed. An image of a bed of small multicolored flowers will always compress much less than a picture of a uniformly colored wall.



Zig-Zag Ordering and Entropy Encoding (Zero Run-Length and Huffman Coding)

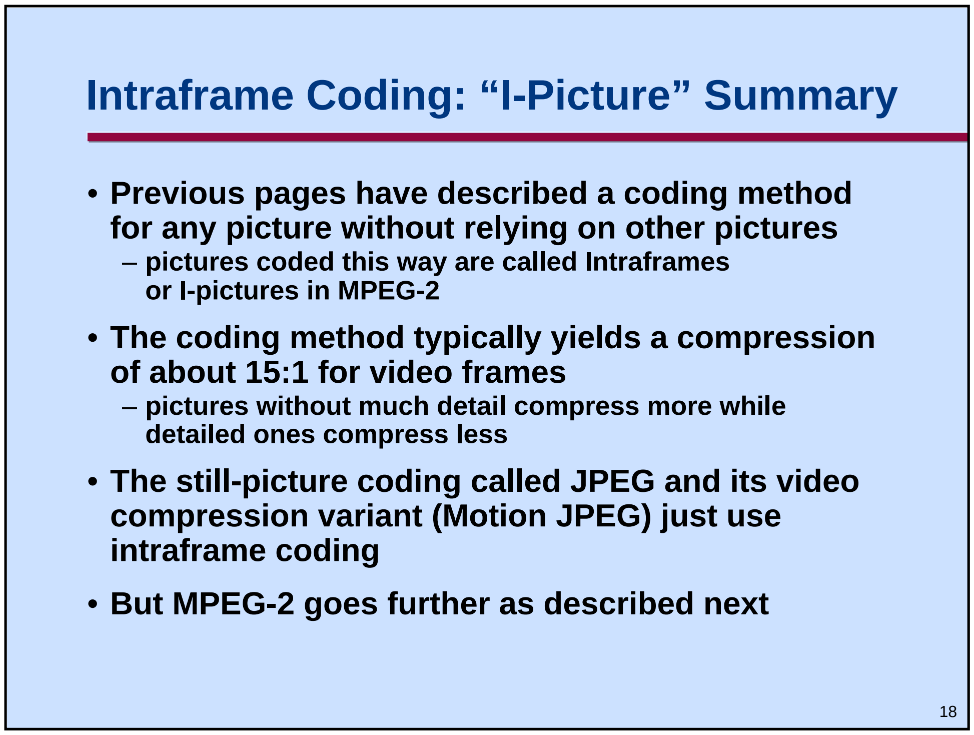


- Values near upper left are encoded most precisely because they are most important visually
- Zeroes are most likely to be bunched up toward end of string and are not important visually
- Consecutive zero runs are coded very compactly
- Only 10 numbers instead of 64 need to be stored or transmitted in this example of one 8x8 pixel block

17

This step in the image compression process starts with first reading the transformed and quantized numbers in the 8x8 block in a zig-zag manner starting with the DC term in the upper left and ending with the term in the lower right, in other words, going from lowest to highest “frequency” at least approximately. The most important numbers will now be at the head of the list and the zero terms are more likely to be toward the end of the list, and hopefully—to get good compression—there will be a lot of zeroes in an average quantized block. The first term (the DC term) is often the largest and often the most relevant. Putting all the zeroes away from the front means that they can be treated in a compact manner, much like using scientific notation for very large or very small numbers. Because the string of quantized numbers is likely to have long runs of zeroes, these can be efficiently encoded by a run-length encoder, which simply gives the number of zeroes before a non-zero number. Huffman coding is used to achieve further compression.

MPEG-2 offers two ordering patterns: the zig-zag one shown above and a second one which has a modified zig-zag pattern. The second one is better for coding interlaced format, in that it improves compression measurably.



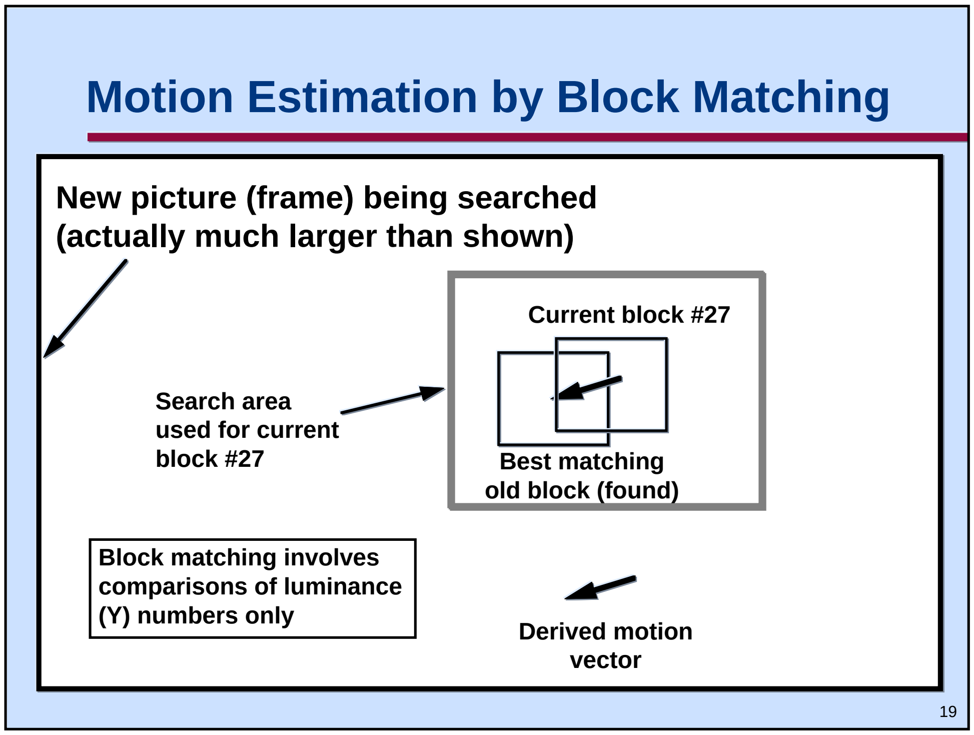
Intraframe Coding: “I-Picture” Summary

- Previous pages have described a coding method for any picture without relying on other pictures
  - pictures coded this way are called Intraframes or I-pictures in MPEG-2
- The coding method typically yields a compression of about 15:1 for video frames
  - pictures without much detail compress more while detailed ones compress less
- The still-picture coding called JPEG and its video compression variant (Motion JPEG) just use intraframe coding
- But MPEG-2 goes further as described next

18

The previous pages have described a method for coding and compressing individual pictures (frames) in a video sequence. This is called intraframe coding and results in what are called I-pictures in the MPEG-2 standard. This method is also one used in coding still pictures per the JPEG specification. Because of round off in FDCT and because of quantization, this is a lossy method; in other words, the decoded picture can be a very good replica of the original, but not a perfect one. In JPEG, there is also a method for doing lossless coding, which is very important for medical imaging, but the degree of compression achieved is lower than the 15:1 one gets on average using the one just described.

Motion JPEG uses the intrapicture method described in the preceding pages. MPEG-2 goes one step further, as shown next, to gain another factor of 3 or 4:1 in compression.

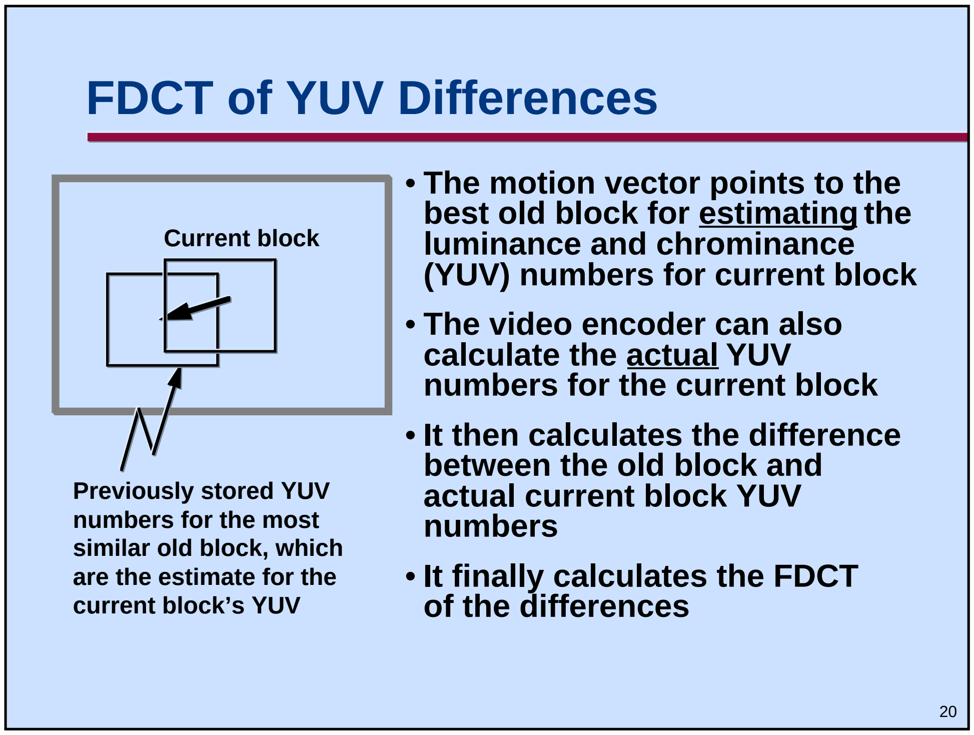


Motion Estimation by Block Matching

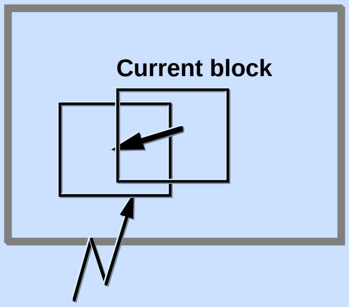
A key part of video coding and compression methods is to take advantage of redundant information in successive groups of pictures as well as within pictures, which is what I-picture coding was all about. To find such redundant information, which should not have to be stored or sent, some video coding schemes, such as MPEG-2, use block matching between successive frames.

The principle works as follows: Instead of just FDCT-coding the numbers for a block as was done for each I-picture block, why not first try to find whether there was already an identical, or at least very similar, block in the previous frame. So for the current block #27 above, a search is made in the previous frame near the location of block #27 (which is in the current frame) to find the most similar block near it in that previous frame. If there was no motion, the location in the previous frame will be the same as where #27 is in the current frame. If there was motion, the block being searched for may turn up where shown. (These searches are, of course, done in a holding memory for the earlier frame, since we are doing everything digitally here). As we'll see, it's not necessary to find an identical block to achieve further compression. But if one were found, it would be potentially much more economical in terms of bytes stored or bits sent, to just refer to the location where an identical block already exists earlier in the video sequence (which was stored).

Above, the best matching block to the current one (#27) is identified by a "motion vector." Vectors on a two-dimensional surface are simple to code from a reference point (say the lower left corner of block #27)—it just takes two numbers (x and y coordinates). The blocks being used for comparison are macro-blocks (16x16 pixels) which contain 256 numbers for luminance alone (before compression). So, having to send a motion vector only, provides a compression of 256/2 or 128:1. But, of course the block is compressed, and the motion vector can't be compressed much or at all—and there is the matter that an identical block often does not exist. But is there compression to be gained in pursuing this approach?



FDCT of YUV Differences



Previously stored YUV numbers for the most similar old block, which are the estimate for the current block's YUV

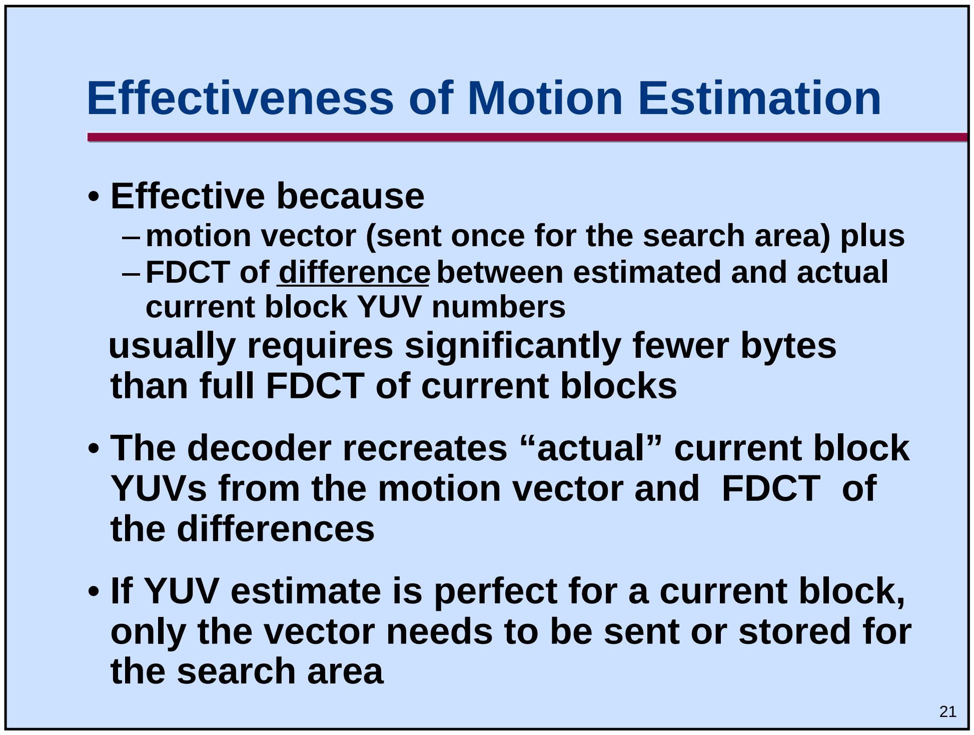
- The motion vector points to the best old block for estimating the luminance and chrominance (YUV) numbers for current block
- The video encoder can also calculate the actual YUV numbers for the current block
- It then calculates the difference between the old block and actual current block YUV numbers
- It finally calculates the FDCT of the differences

20

This slide describes the block-based coding method which takes advantage of the fact that previously processed frames (pictures) already contain much of the information in the current frame.

- The motion vector for an area is simply the relative address of the memory location holding the YUV numbers of most similar block in an earlier frame to the current block being processed. These previously stored YUV numbers become the estimate of YUV numbers for the current block.
- The encoder also calculates the actual YUV numbers for the current block. (Why not just use these instead of estimating, as done in the previous step? Read on).
- The encoder then calculates the differences between the actual and estimated YUV numbers. (The reason for doing this is the presumption that the differences can be coded more efficiently (i.e., with fewer bytes) than simply coding the actual YUV numbers. The validity of the assumption is treated on the next two slides.)
- The encoder finally calculates the FDCT of the YUV differences.

All this sounds like a lot of extra work for no apparent gain. There is, in fact, a gain to be achieved in compression, and the effectiveness of this extra work to that end is discussed next.



Effectiveness of Motion Estimation

- **Effective because**
  - **motion vector (sent once for the search area) plus**
  - **FDCT of difference between estimated and actual current block YUV numbers**

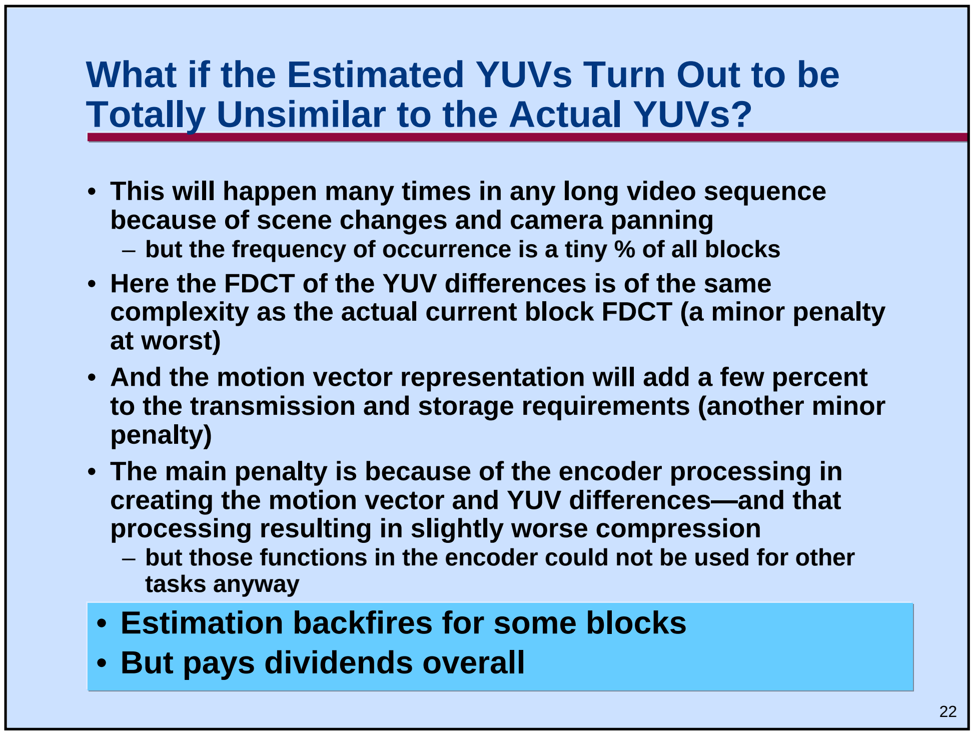
  usually requires significantly fewer bytes than full FDCT of current blocks
- **The decoder recreates "actual" current block YUVs from the motion vector and FDCT of the differences**
- **If YUV estimate is perfect for a current block, only the vector needs to be sent or stored for the search area**

21

The block-based technique of eliminating storage space and transmission requirements because of redundancy between successive frames, is referred to as motion estimation. Its effectiveness rests on the assumption that having to store or send the motion vector and the FDCT of the difference between actual and estimated YUV numbers requires fewer bytes than storing or sending the actual FDCT of current blocks. More is said about that on the next slide.

The decoder recreates the actual current block YUV numbers from the motion vector and the DCT of the differences. To do decoding, it first calculates the inverse DCT (IDCT) of the differences, which yields the YUV difference numbers for a block. Since it has kept all the blocks for several previous frames, it uses the motion vector to find the nearest similar block to the one it needs, and does not yet have. It adds the YUV differences to the earlier frame's block YUV numbers and the result is the actual YUV numbers of the current block. This process is repeated for all blocks in the frame during the decoding process.

For some blocks, the YUV differences are zero, and for those, all the decoder needs to do is retrieve the actual YUVs from the memory location identified by the motion vector, since that is just a relative memory address from the memory location where the current block's YUV numbers are to be stored. Note that decoding is only done when the video is to be played out in real time (i.e. displayed on a monitor). If the video is sent out through a network to a remote site for digital storage on a non-destructive medium such as video disk or (digital) tape, it is stored in coded form. If there is no digital storage capability, as in a typical home set-up for digital video reception from a satellite, the receiver decodes the information, transforms it to NTSC or PAL, for example, and it is then stored on a VHS video (analog) recorder tape.



What if the Estimated YUVs Turn Out to be Totally Unsimilar to the Actual YUVs?

- This will happen many times in any long video sequence because of scene changes and camera panning
  - but the frequency of occurrence is a tiny % of all blocks
- Here the FDCT of the YUV differences is of the same complexity as the actual current block FDCT (a minor penalty at worst)
- And the motion vector representation will add a few percent to the transmission and storage requirements (another minor penalty)
- The main penalty is because of the encoder processing in creating the motion vector and YUV differences—and that processing resulting in slightly worse compression
  - but those functions in the encoder could not be used for other tasks anyway

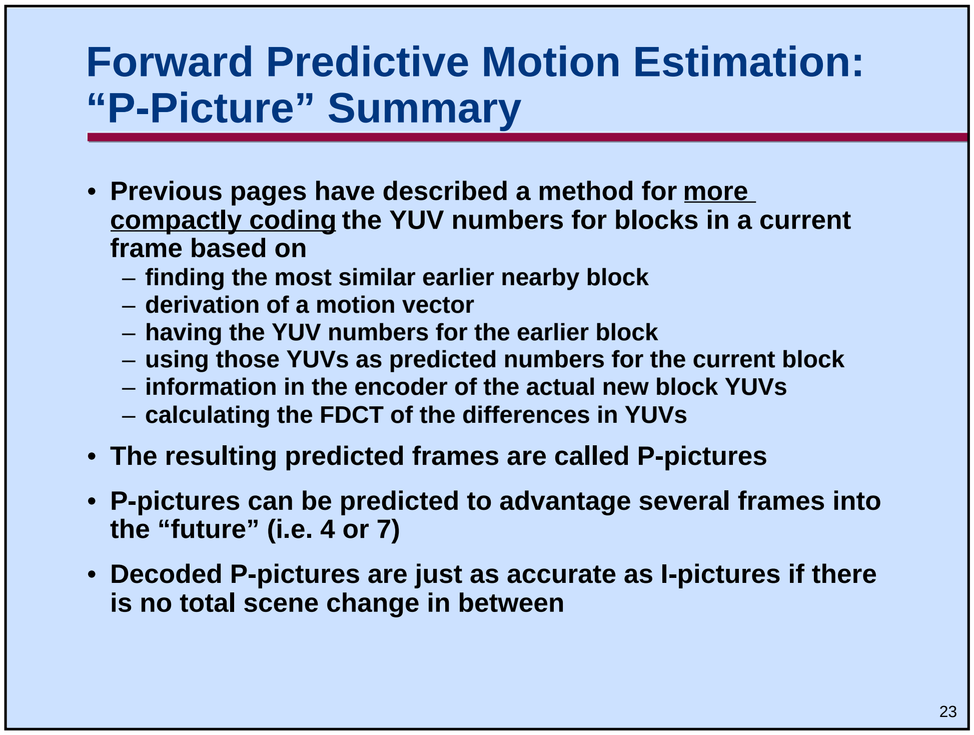
- **Estimation backfires for some blocks**
- **But pays dividends overall**

22

A person going through the motion estimation process is likely to wonder what happens if there is an abrupt scene change because of a change of camera, as in a sports video sequence. In such a situation, all past blocks will be totally different from the blocks in the current frame. Even the “most similar”—which will still be found because of the algorithm used—will have no meaningful similarity. So, does the whole process fall apart? The answer is no. But compression for that current frame will be a little worse than if the frame had been treated as an I-picture. The FDCT of the YUV differences will be of the same complexity as the FDCT of the actual current block’s YUV numbers. But the motion vector will have to be sent or transmitted, which is a waste of a fairly small number of bytes. So in a storage and transmission sense, the motion estimation encoding process will have reduced compression in a minor way, compared to the I-picture method. And all the processing for the motion vector calculation and calculating the differences of the YUV numbers will have been for nothing. But that is a trivial issue for an embedded processor which has nothing else to do anyway. And the FDCT calculation for the YUV differences is the same as for I-pictures, so that is not a waste.

The bottom line is that motion estimation backfires for some blocks in a frame, or even for entire frames. But overall, it results in greater compression, as shown next.

As an encoder design side note, the same circuit and method used for the block-based motion estimation can also be used to code the I-pictures discussed earlier. All that is required is one bit of control information to disable the search for a motion vector, and to set the “found” YUVs to zero, and then to run through the process. So there is no duplication of circuitry to do I-pictures.



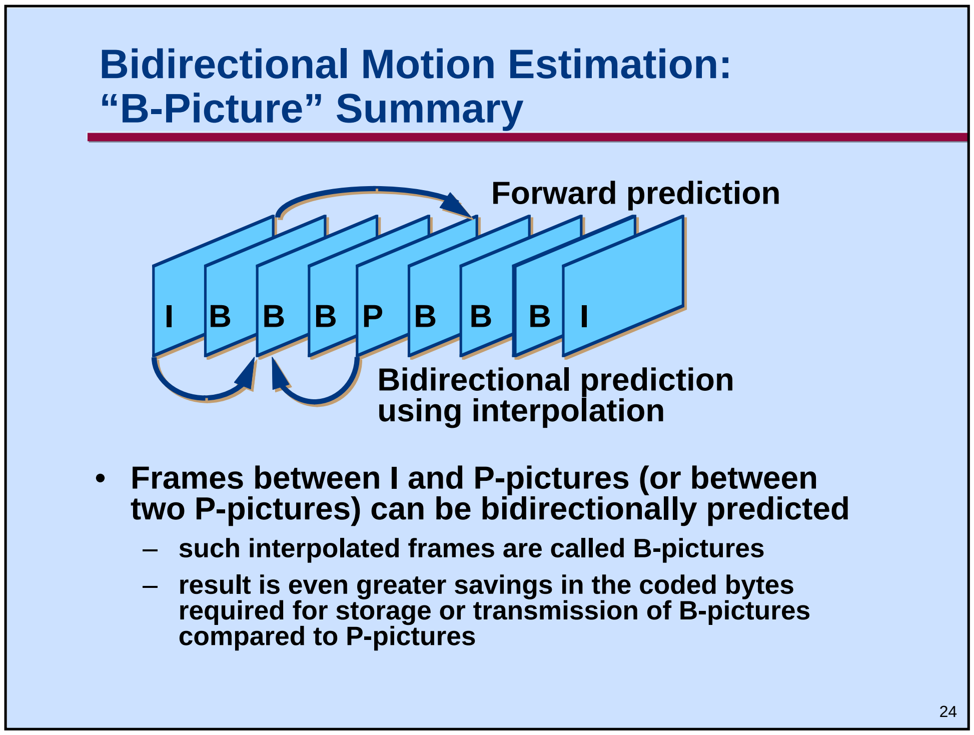
Forward Predictive Motion Estimation: “P-Picture” Summary

- Previous pages have described a method for more compactly coding the YUV numbers for blocks in a current frame based on
  - finding the most similar earlier nearby block
  - derivation of a motion vector
  - having the YUV numbers for the earlier block
  - using those YUVs as predicted numbers for the current block
  - information in the encoder of the actual new block YUVs
  - calculating the FDCT of the differences in YUVs
- The resulting predicted frames are called P-pictures
- P-pictures can be predicted to advantage several frames into the “future” (i.e. 4 or 7)
- Decoded P-pictures are just as accurate as I-pictures if there is no total scene change in between

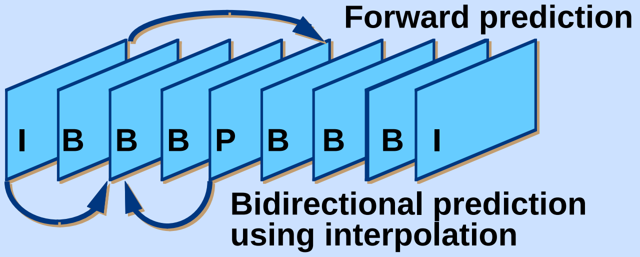
23

The last few pages have described the block-based method of coding more compactly the YUV numbers in a video sequence than is possible with just the I-picture (intrapicture) method described earlier. The resulting frames are called P-pictures, and they can be coded to gain compression advantage several frames into the future. MPEG-2 actually allows coding 128 frames into the future, but for a typical video sequence there is a diminishing point of return on the effectiveness of the process after 10 or 15 frames. (Note that “the future” is just a reference term. The encoder must store 128 frames before sending the first frame in such a sequence, and in fact for the entire film clip, if it is to see 128 frames ahead. So it has already seen and recorded “the future” because it has delayed the sending process by whatever the maximum number of frames ahead P-pictures are to be coded. Remember that there are only 25 frames per second in a PAL TV system so even storing 15 frames means the video sequence is delayed by 15/25 seconds or 600 ms—which is unacceptable for video conferencing, for example, but fine for TV broadcasting.)

The decoded P-pictures are just as good as decoded I-pictures, but are compressed two or three times more, except when there is a scene change in the sequence—in which case they are compressed less. The net gain is a function of how frequently scene changes occur, which varies by type of film clip. So using the P-picture method saves memory for storing a film sequence and saves a proportional amount of transmission bandwidth. But the encoder and decoder must have more complex and faster circuitry and there is a price paid in terms of delay of the video sequence which affects editing, caused by the fact that only an I-picture can be an editing entry point. An I-picture is the reference for the initial P-picture in a group of pictures (although any P-picture can then be a reference for several others). So, if there is an attempt to gain compression by using many P-pictures without any I-pictures in between, editing complexity and process time can easily become unacceptable.



Bidirectional Motion Estimation: "B-Picture" Summary

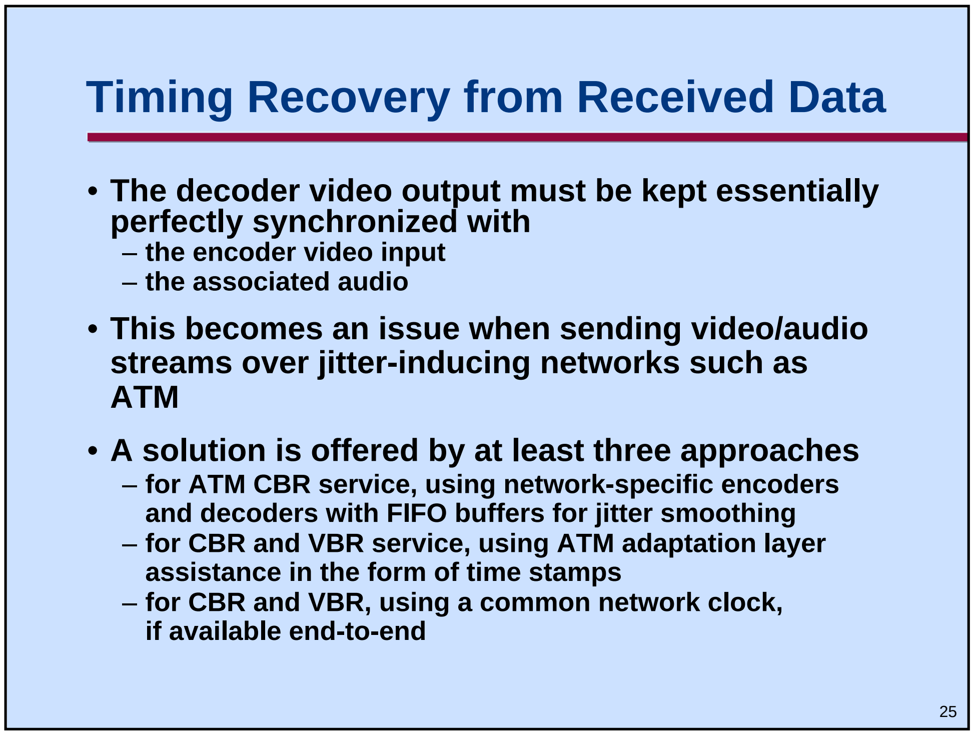


- Frames between I and P-pictures (or between two P-pictures) can be bidirectionally predicted
  - such interpolated frames are called B-pictures
  - result is even greater savings in the coded bytes required for storage or transmission of B-pictures compared to P-pictures

24

There is an even greater compression possibility using block-based techniques than just with P-pictures and I-pictures. By using reference frames from both the past and future direction, it is possible to find more nearly identical blocks than from one direction alone. For example, if there has been a scene change, then the P-picture method requires more bytes than doing everything from the beginning (I-picture coding). But, if a frame from the future direction is used, then the P-picture method would work well. The approach of coding pictures using past and future reference points is called bidirectional extrapolation, and the results are called B-pictures. There are several variations to the theme beyond the description two sentences back. The B-picture result is an even greater savings in storage and transmission requirements than with P-pictures.

In an average video sequence, P and B-picture coding collectively improve compression by a factor of 3:1 or 4:1 on top of the 15:1 typically obtained from intracoding (I-pictures) alone.



Timing Recovery from Received Data

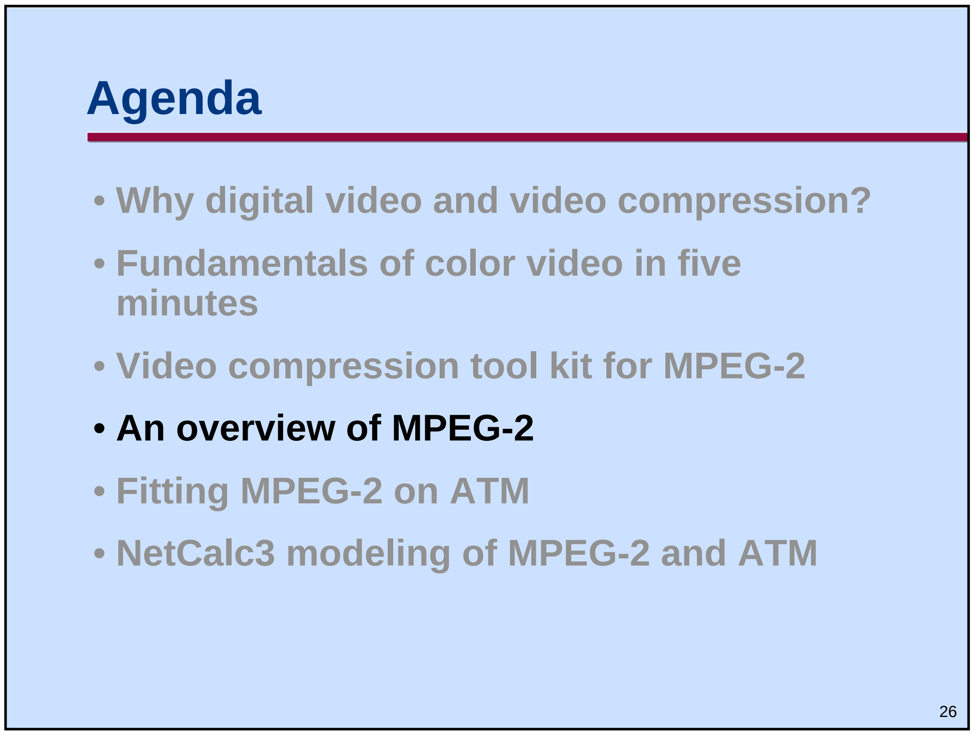
- **The decoder video output must be kept essentially perfectly synchronized with**
  - **the encoder video input**
  - **the associated audio**
- **This becomes an issue when sending video/audio streams over jitter-inducing networks such as ATM**
- **A solution is offered by at least three approaches**
  - **for ATM CBR service, using network-specific encoders and decoders with FIFO buffers for jitter smoothing**
  - **for CBR and VBR service, using ATM adaptation layer assistance in the form of time stamps**
  - **for CBR and VBR, using a common network clock, if available end-to-end**

25

The decoder video output must be kept synchronized with the encoder output to avoid overruns or underruns in the receiving (decoder) buffer memory. Failure to maintain synchronization causes severe visual disruptions of a video sequence. Associated audio must also be kept synchronized with the received video sequence or there will be “lip-synch” problems like movies from the 1920s often used to have.

Jitter-inducing networks, notably ATM, are a particular problem in these regards, and force implementations of solutions.

- The first-in-first out (FIFO) buffer solution is easy to implement and reduces short-term variations in received data. But it does not solve the problem fully.
- The use of ATM adaptation layer (AAL) assistance in the form of time stamps allows gradual slewing of the phase-locked loop in the decoder in the right direction, independently of any ATM network clock. Each decoder has its own oscillator which free-runs at near the expected encoder frequency. A phase-locked loop is also built into the decoder with the idea of phase-locking the decoder’s oscillator to that of the encoder. But if a common network clock is not available, how can the error signal to slew the phase-locked loop be derived? The answer is with the AAL time stamps. This solution is used in conjunction with FIFO buffers.
- For cases where there is a common network clock, synchronization is easier. It appears, however, that in most situations this will not be possible.

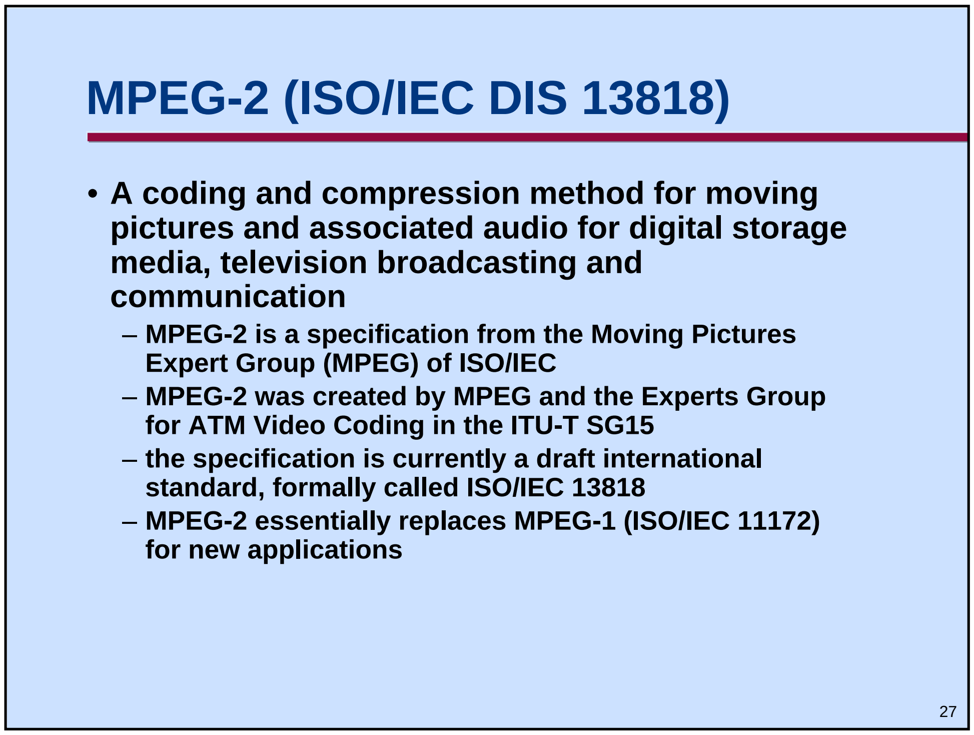


Agenda

- Why digital video and video compression?
- Fundamentals of color video in five minutes
- Video compression tool kit for MPEG-2
- **An overview of MPEG-2**
- Fitting MPEG-2 on ATM
- NetCalc3 modeling of MPEG-2 and ATM

26

This section attempts to describe MPEG-2 by bringing all the building blocks, described in the last section, together. The approach used here is different than most descriptions of MPEG-2—in fact, almost the reverse of many of them. If this one does not work for the reader or workshop attendee, other descriptions in the references at the end will help. In learning about MPEG-2, it may prove worthwhile to approach the subject a few different ways. That’s what led to the approach used in this workshop.



MPEG-2 (ISO/IEC DIS 13818)

- A coding and compression method for moving pictures and associated audio for digital storage media, television broadcasting and communication
  - MPEG-2 is a specification from the Moving Pictures Expert Group (MPEG) of ISO/IEC
  - MPEG-2 was created by MPEG and the Experts Group for ATM Video Coding in the ITU-T SG15
  - the specification is currently a draft international standard, formally called ISO/IEC 13818
  - MPEG-2 essentially replaces MPEG-1 (ISO/IEC 11172) for new applications

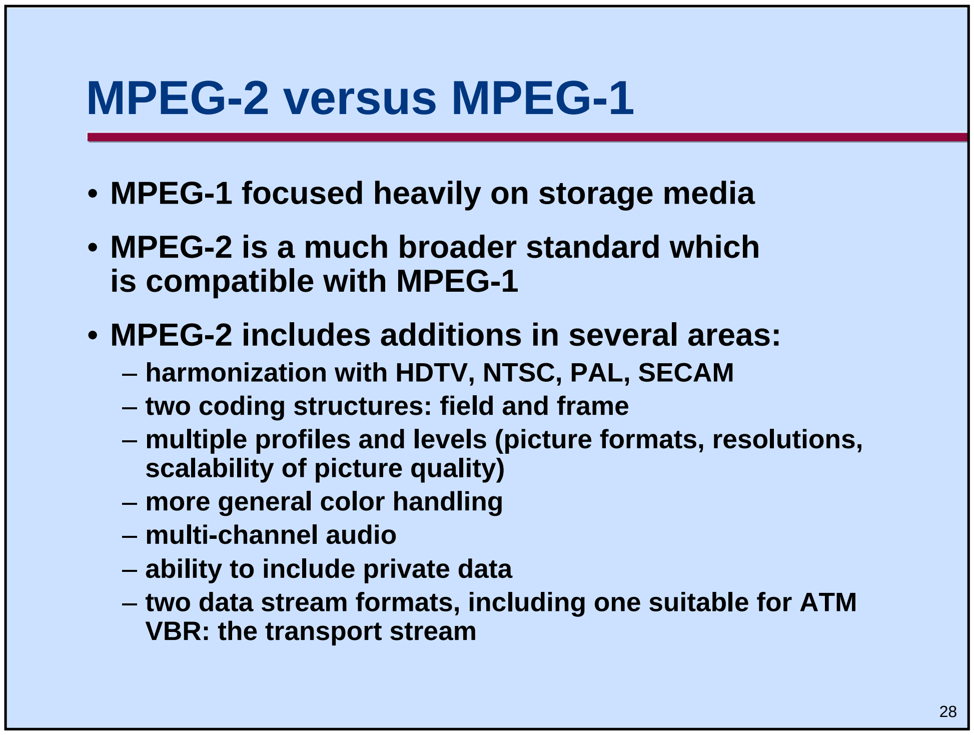
27

This slide defines MPEG-2, presents where it comes from, and shows its formal name.

Note that while MPEG-2 largely replaces MPEG-1 for new applications, MPEG-4 is a new proposed standard for very low bit rate (10 kbit/s to 64 kbit/s) audio/visual coding. MPEG-2 is targeted at normal and high definition television quality video (2 Mbit/s to over 60 Mbit/s). To achieve greater compression for very low bandwidth applications, MPEG-4 is probably going to be object-oriented while MPEG-2 uses a block-based compression scheme. The coding method for MPEG-4 is not finalized as of March 1996.

Below is a list of the common coding standards:

Standard	Bit Rate Range	Application
MPEG-4	10 to 64 kbit/s	Video conferencing over one voice channel
H.261	64 kbit/s to 2 Mbit/s	High quality video conferencing (over ISDN)
MPEG-1	1.5 Mbit/s+	VHS-quality TV (for NTSC, PAL, SECAM)
MPEG-2	2 Mbit/s to 60 Mbit/s+	NTSC, PAL, SECAM, HDTV-quality TV

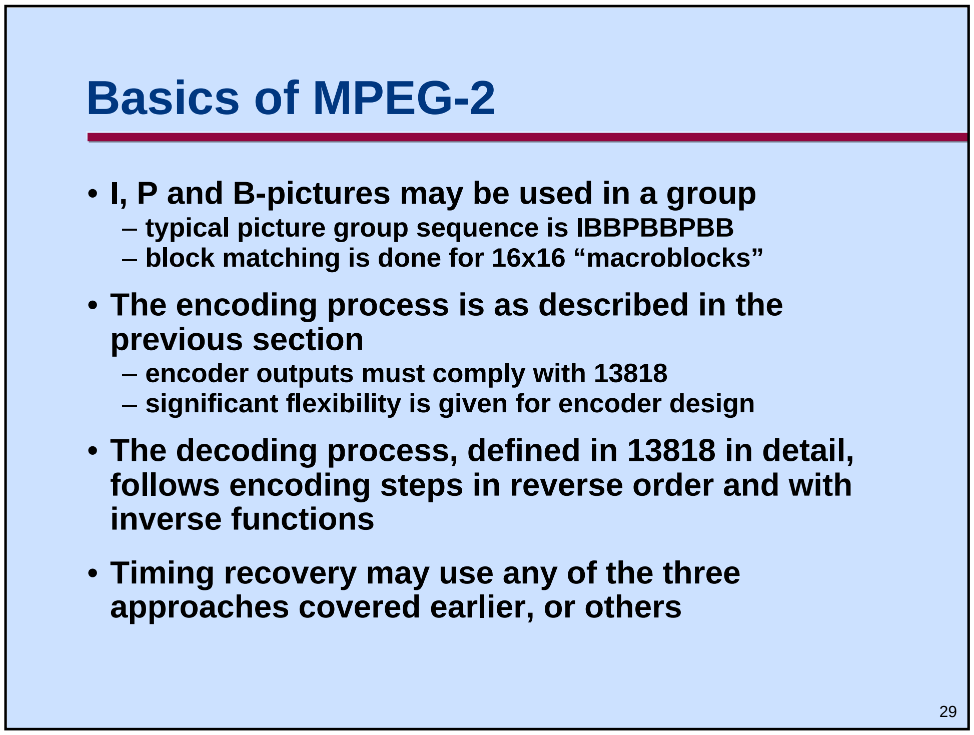


MPEG-2 versus MPEG-1

- MPEG-1 focused heavily on storage media
- MPEG-2 is a much broader standard which is compatible with MPEG-1
- MPEG-2 includes additions in several areas:
  - harmonization with HDTV, NTSC, PAL, SECAM
  - two coding structures: field and frame
  - multiple profiles and levels (picture formats, resolutions, scalability of picture quality)
  - more general color handling
  - multi-channel audio
  - ability to include private data
  - two data stream formats, including one suitable for ATM VBR: the transport stream

28

MPEG-2 is a more comprehensive standard than MPEG-1 and—importantly for terrestrial networks and this workshop—was designed for compatibility with ATM.



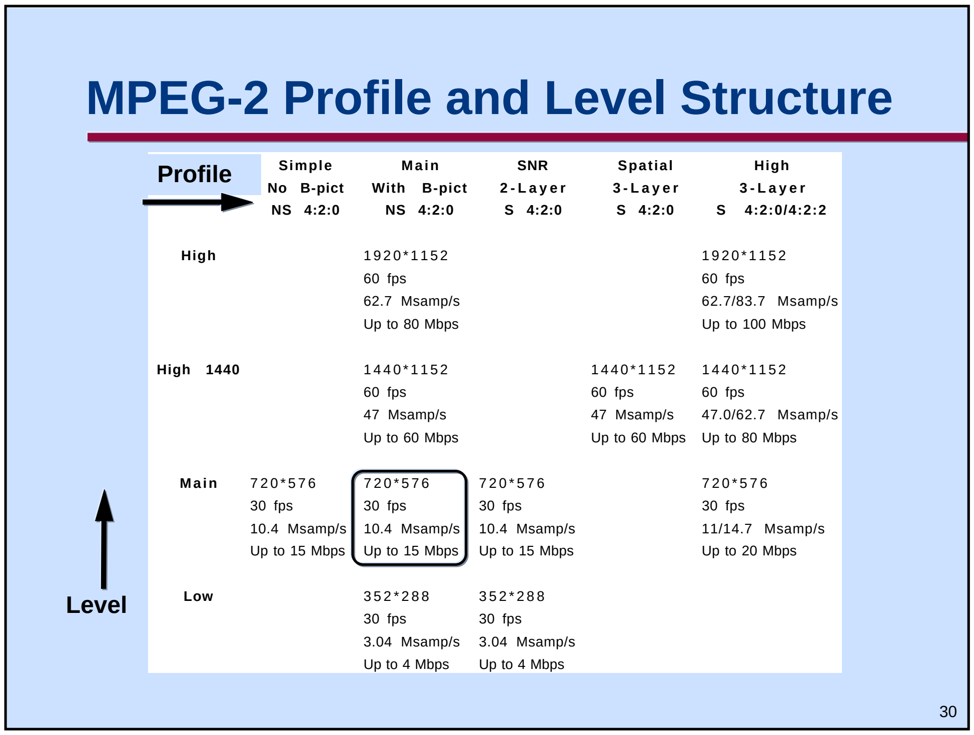
Basics of MPEG-2

- I, P and B-pictures may be used in a group
  - typical picture group sequence is IBBPBBPBB
  - block matching is done for 16x16 “macroblocks”
- The encoding process is as described in the previous section
  - encoder outputs must comply with 13818
  - significant flexibility is given for encoder design
- The decoding process, defined in 13818 in detail, follows encoding steps in reverse order and with inverse functions
- Timing recovery may use any of the three approaches covered earlier, or others

29

Because MPEG-2 is built using the theory briefly covered in the previous section, its basics can be run through quickly.

- I,P and B-pictures can be used, though the encoder can decide at any time which are used.
  - Note that the typical group of pictures has about nine frames in it, even though the standard allows up to 128. The reasons are due to the trade-offs discussed earlier.
  - Block matching (to derive motion vectors) is done for 16x16 pixel macroblocks, which contain four 8x8 blocks, in order to save processing cycles and bytes for storage and transmission.
- The encoding process is not defined in the standard—only its results are.
- The decoding process is defined in detail.
- The method used for timing recovery is also not defined rigidly. Any or all of the three methods described earlier can be used—bearing in mind that the choice impacts the design of the encoder and decoder. Because MPEG-2 will be used more for broadcasting than one-to-one communications, encoders are viewed as being much less cost-constrained than decoders in the 13818 standard.



MPEG-2 Profile and Level Structure

Level \ Profile	Simple No B-pict NS 4:2:0	Main With B-pict NS 4:2:0	SNR 2-Layer S 4:2:0	Spatial 3-Layer S 4:2:0	High 3-Layer S 4:2:0/4:2:2
High		1920*1152 60 fps 62.7 Msamp/s Up to 80 Mbps			1920*1152 60 fps 62.7/83.7 Msamp/s Up to 100 Mbps
High 1440		1440*1152 60 fps 47 Msamp/s Up to 60 Mbps		1440*1152 60 fps 47 Msamp/s Up to 60 Mbps	1440*1152 60 fps 47.0/62.7 Msamp/s Up to 80 Mbps
Main	720*576 30 fps 10.4 Msamp/s Up to 15 Mbps	720*576 30 fps 10.4 Msamp/s Up to 15 Mbps	720*576 30 fps 10.4 Msamp/s Up to 15 Mbps		720*576 30 fps 11/14.7 Msamp/s Up to 20 Mbps
Low		352*288 30 fps 3.04 Msamp/s Up to 4 Mbps	352*288 30 fps 3.04 Msamp/s Up to 4 Mbps		

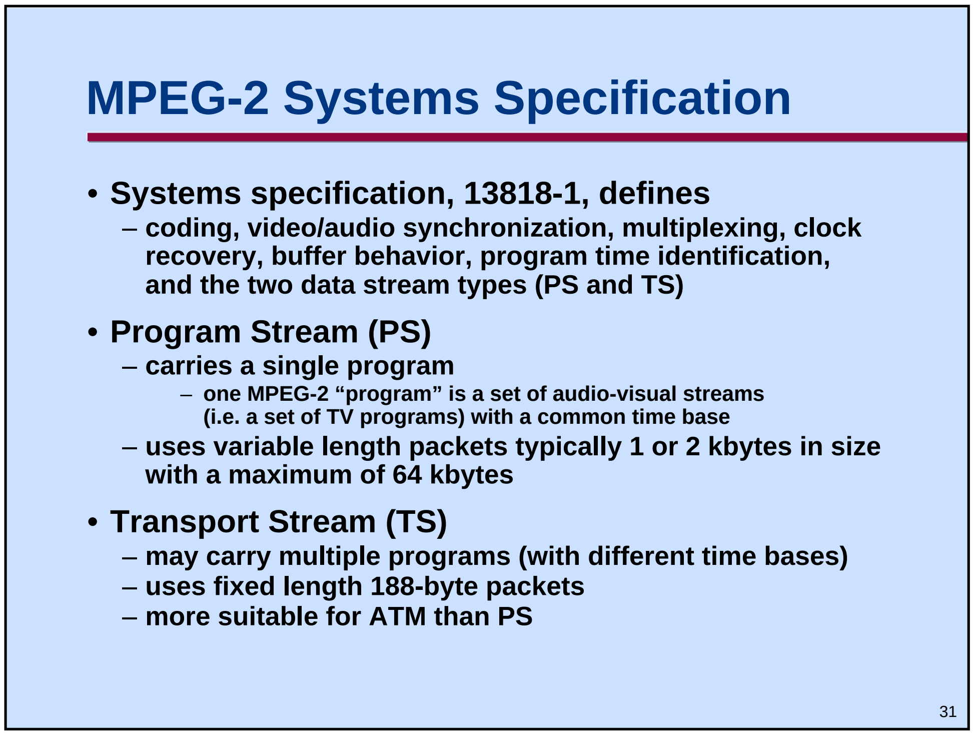
30

Profiles and levels provide a means of defining subsets of the syntax and semantics of the video part of the 13818 standard, and thereby define the decoder capabilities to decode a particular bitstream. For details, such as 4:2:2 chrominance format, refer to [1] and [7].

- A profile is a defined subset of the entire bitstream syntax of 13818.
- A level is a set of constraints imposed on parameters in the bitstream.

The most common profile and level is outlined on the slide, and it applies to normal definition television. It defines the maximum resolution for the picture and also the highest bit rate (after MPEG-2 encoding) to be 15 Mbit/s. In other words, if the transmission medium can carry a peak of 15 Mbit/s, the picture quality will be excellent (for that profile and level). This does not mean that the average bit rate is going to be 15 Mbit/s; in fact, it will typically be much lower, perhaps 4 Mbit/s. It also does not mean that the picture will not necessarily be good if the peak bit rate were to be restricted to, say, 9 Mbit/s. But it does mean that the picture quality will not be excellent for all video sequences under that condition. Note that the MPEG-2 coding method automatically copes with reductions in peak bandwidth in a very elegant way: It starts to reduce quality of individual frames in the least perceptible way to the viewer, and only causes visually annoying effects, such as frame freezing, when nothing else is possible with its coding method or implementation for the given maximum bandwidth.

The 13818 standard does not specify anything except peak bit rate to achieve excellent quality for each entry in the slide. Measured data beyond the specification is required for a trade-off of picture quality versus peak bit rate [7]. For network design purposes at Nortel, that information, gathered from various articles and tests in the public domain, are embedded into the network design tool described in the next section.



MPEG-2 Systems Specification

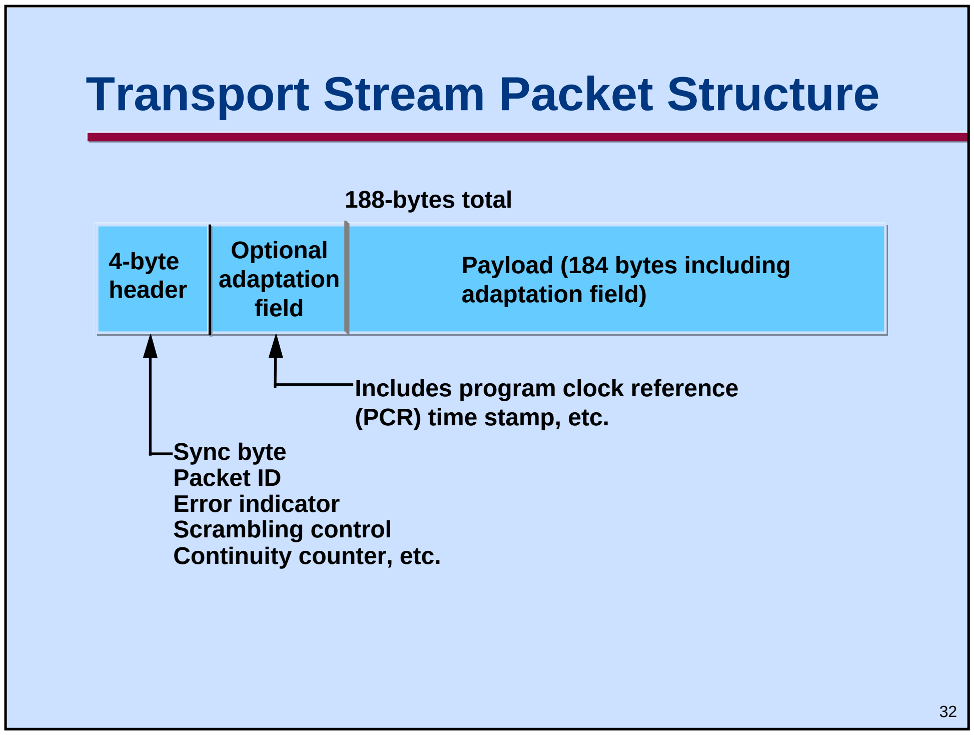
- Systems specification, 13818-1, defines
  - coding, video/audio synchronization, multiplexing, clock recovery, buffer behavior, program time identification, and the two data stream types (PS and TS)
- Program Stream (PS)
  - carries a single program
    - one MPEG-2 “program” is a set of audio-visual streams (i.e. a set of TV programs) with a common time base
  - uses variable length packets typically 1 or 2 kbytes in size with a maximum of 64 kbytes
- Transport Stream (TS)
  - may carry multiple programs (with different time bases)
  - uses fixed length 188-byte packets
  - more suitable for ATM than PS

31

The 13818 standard, which is well over 500 pages long, has four major parts:

- Part 1, the System Specification, defines the items listed above, and among them defines the data stream types, of which the Transport Stream (TS) is particularly important for ATM and the purposes of this workshop.
- Part 2, called Video, specifies coded representation of video data and the decoding process required to reconstruct pictures.
- Part 3, called Audio, specifies coded representation of audio data and the decoding process required to reconstruct audio signals.
- Part 4, Compliance Testing, specifies how tests can be designed to verify whether bitstreams and decoders meet the requirements specified in parts 1, 2, and 3. Note that by defining bitstream compliance, encoder compliance is fully defined, even though the 13818 standard says very little about encoders.

The two most relevant points of 13818 for network design are the two streams PS and TS—and TS especially because it was designed to be as compatible as possible with ATM. The PS data structure, which is the same as for MPEG-1 in essence, is well-suited for storage media and software control because of its long, variable length packets. But TS, with its cell-like fixed length packets, fits better into the ATM cell structure as shown next. The MPEG-2 specification defines how data can be moved between TS and PS. There is an underlying 13818 packet structure called the packetized elementary stream (PES) which is the linchpin between PS and TS.



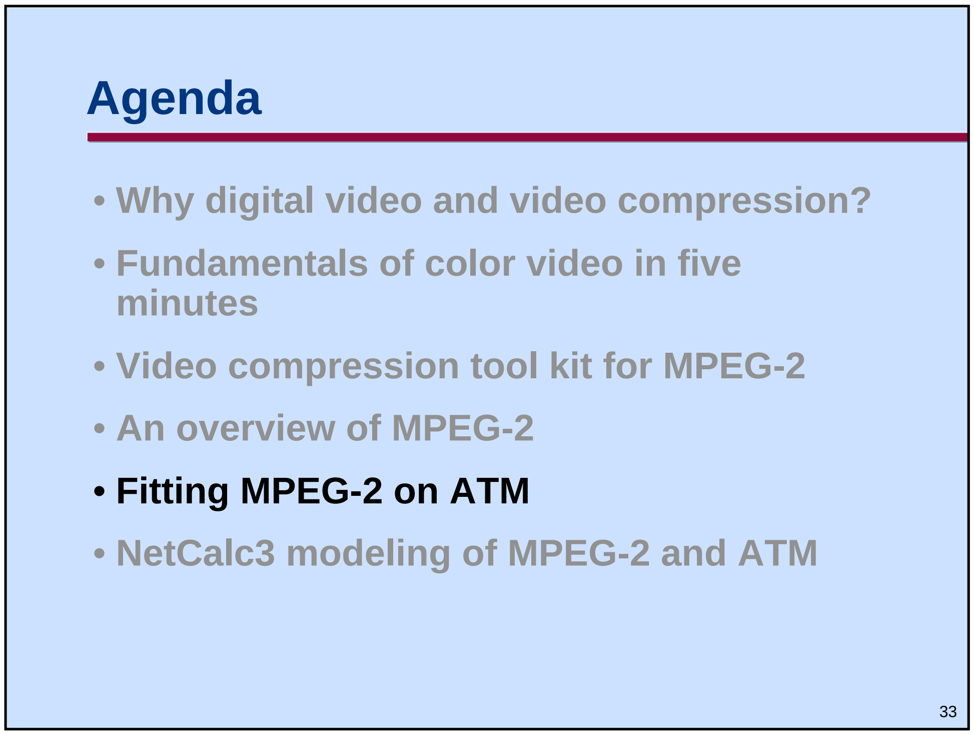
Transport Stream Packet Structure

188-bytes total
4-byte header
Optional adaptation field
Payload (184 bytes including adaptation field)
Includes program clock reference (PCR) time stamp, etc.
Sync byte
Packet ID
Error indicator
Scrambling control
Continuity counter, etc.
32

The TS packet structure is shown above. Anyone wanting greater detail should look at ISO 13818, Part 2, Section 2.4 and Annex F.

Each video frame (picture) creates a variable length PES packet, which may be as long as 62 kbytes even for regular definition television, if the frame is an I-picture with a complex scene. The PES packets are chopped up and the pieces are fitted inside the payload of TS packets, as shown above. Therefore, each frame in a video sequence produces one or many TS packets. With TS, the packets in a group of pictures (GOP), which may be nine frames, for example, is sent or stored so that they start with the I-picture, of which there is but one, and are followed by all the P-pictures and finally by all the B-pictures. So they are sent or stored out of sequence. The reason is that the I and P-pictures are needed first by the decoder, and because they are the most important information in a GOP. If the peak bandwidth required is not sufficient to accommodate the data in a GOP, the B-pictures are thrown out first, and the I-picture last. The decoder puts the pictures back in order after decoding.

MPEG-2 encoders are designed to encode pictures in a GOP in such a way that maximum available bandwidth or memory space is taken into account on a continuous basis. This allows the use of ATM’s CBR service, which may limit quality of the video sequence if CBR data rate is not set high enough. This will not cause major visual disruption unless it is set way too low. CBR is often used with the intent of trading off video quality against transmission bandwidth. The idea behind using VBR service is a different tradeoff. Because the peak bandwidth requirements with video are sporadic, but fairly predictable, VBR can potentially be more efficient than CBR in an ATM network for video because VBR can accommodate high peaks—albeit, without a guarantee. If VBR occasionally does not deliver the peak required, MPEG-2 copes well.

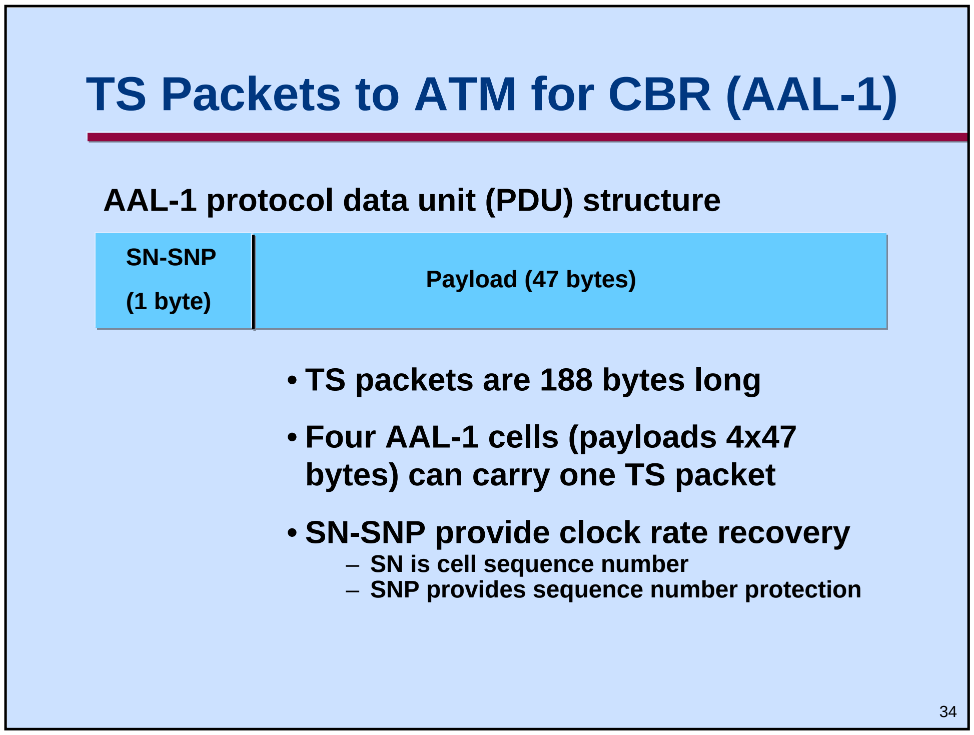


Agenda

- Why digital video and video compression?
- Fundamentals of color video in five minutes
- Video compression tool kit for MPEG-2
- An overview of MPEG-2
- **Fitting MPEG-2 on ATM**
- NetCalc3 modeling of MPEG-2 and ATM

33

This section describes how the MPEG-2 Transport Stream fits on to ATM.



TS Packets to ATM for CBR (AAL-1)

AAL-1 protocol data unit (PDU) structure

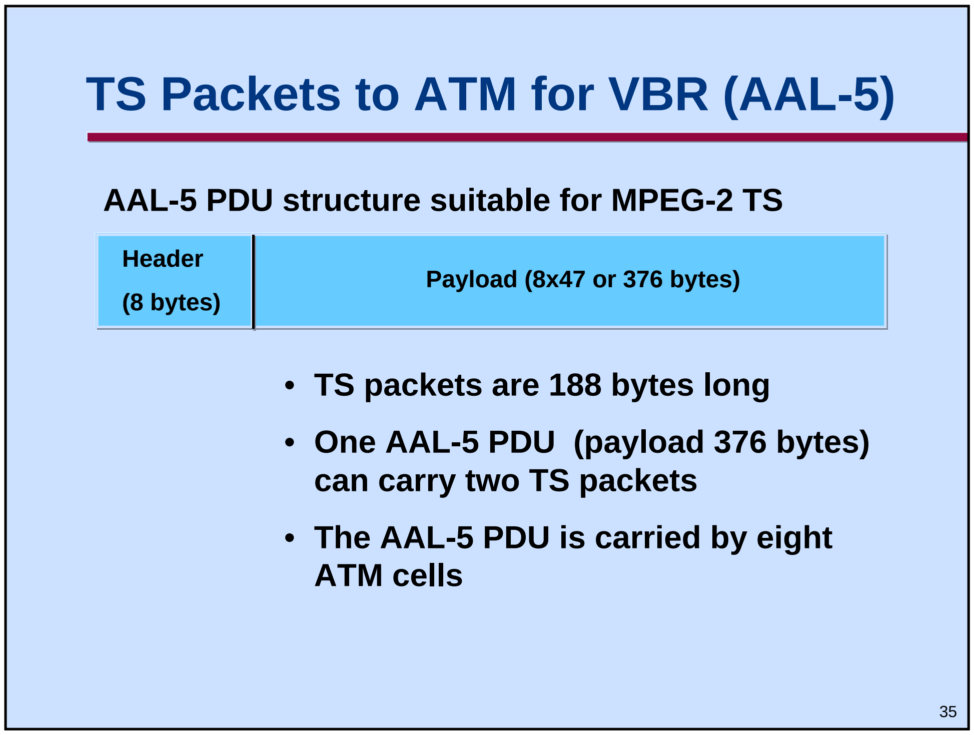
SN-SNP (1 byte)	Payload (47 bytes)

- TS packets are 188 bytes long
- Four AAL-1 cells (payloads 4x47 bytes) can carry one TS packet
- SN-SNP provide clock rate recovery
  - SN is cell sequence number
  - SNP provides sequence number protection

34

Transport Stream packets use two ATM adaptation layers at this time: AAL-1 for CBR service, described here, and AAL-5 for VBR, described on the next chart. Sometime in the future, there will probably emerge a new AAL specifically designed for MPEG-2, but until that happens, AAL-1 and AAL-5 will have to do, and they are adequate.

The TS fixed-length packet size of 188 bytes was chosen such that four AAL-1 cell payloads (4x47 or 188 bytes) can carry one complete TS packet. This still leaves the one byte required for the AAL-1 header per AAL-1 protocol data unit—and, of course, in the ATM layer, which is one below the AAL layer, there are the usual additional 5 bytes of overhead to form a 53-byte ATM cell.



TS Packets to ATM for VBR (AAL-5)

AAL-5 PDU structure suitable for MPEG-2 TS

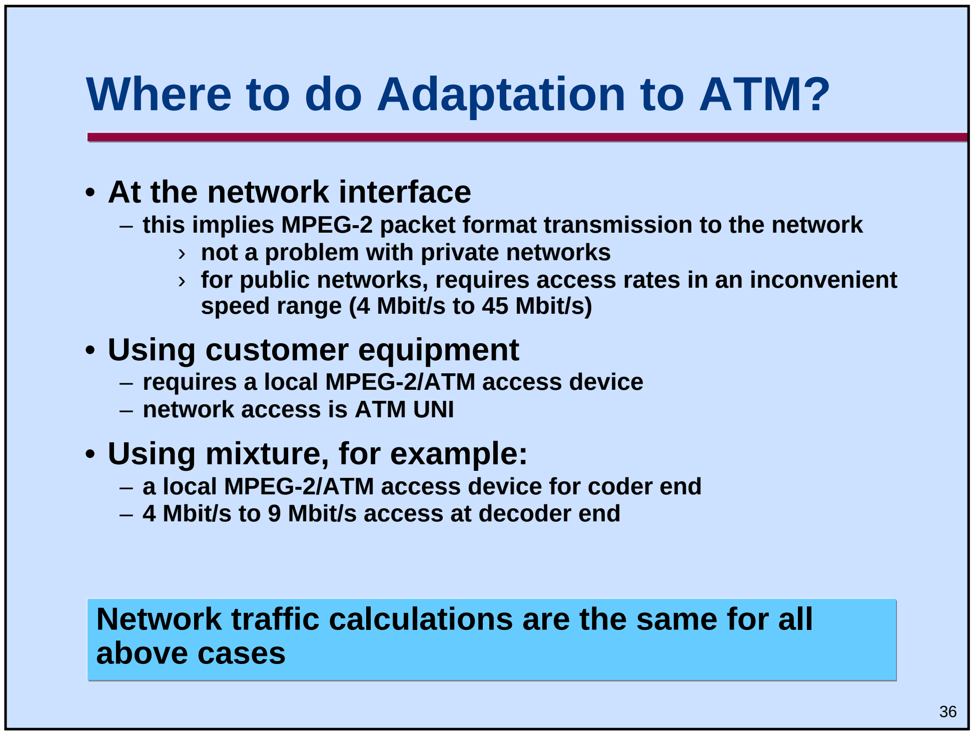
Header (8 bytes)	Payload (8x47 or 376 bytes)

- **TS packets are 188 bytes long**
- **One AAL-5 PDU (payload 376 bytes) can carry two TS packets**
- **The AAL-5 PDU is carried by eight ATM cells**

35

AAL-5 protocol data units are usually much longer than for AAL-1 because the header is 8 bytes long instead of 1 byte. If an AAL-5 payload of 376 bytes is chosen, two complete TS packets can be carried by the AAL-5 PDU, which ends up being 384 bytes long. That PDU can be carried by 8 ATM cells since each cell has a payload of 48 bytes at the ATM layer. So TS packets are easily accommodated by AAL-5. The only major issue with AAL-5 is that it was not really designed for real-time VBR applications such as MPEG-2 video—AAL-2 was supposed to be the AAL for real-time VBR. But AAL-2's definition has dragged on over several years and is not finished as of March 1996. So, the MPEG-2 design community has had to derive other solutions, and the result is that AAL-5 can be used effectively. Refer to [5] for more information on AALs as they apply to MPEG-2.

This and the previous slide provide the means of calculating AAL and ATM overhead as it applies to MPEG-2 Transport Stream packets. Note that because the TS packets are of fixed length and proper size, there are no ATM-layer cells which are not full or at least nearly full. If there are an odd number of TS packets in a video frame, the last AAL-5 PDU simply has a shorter payload resulting in a few wasted bytes. Because of good planning, MPEG-2 Transport Stream packets are carried efficiently by CBR and VBR services in an ATM network.



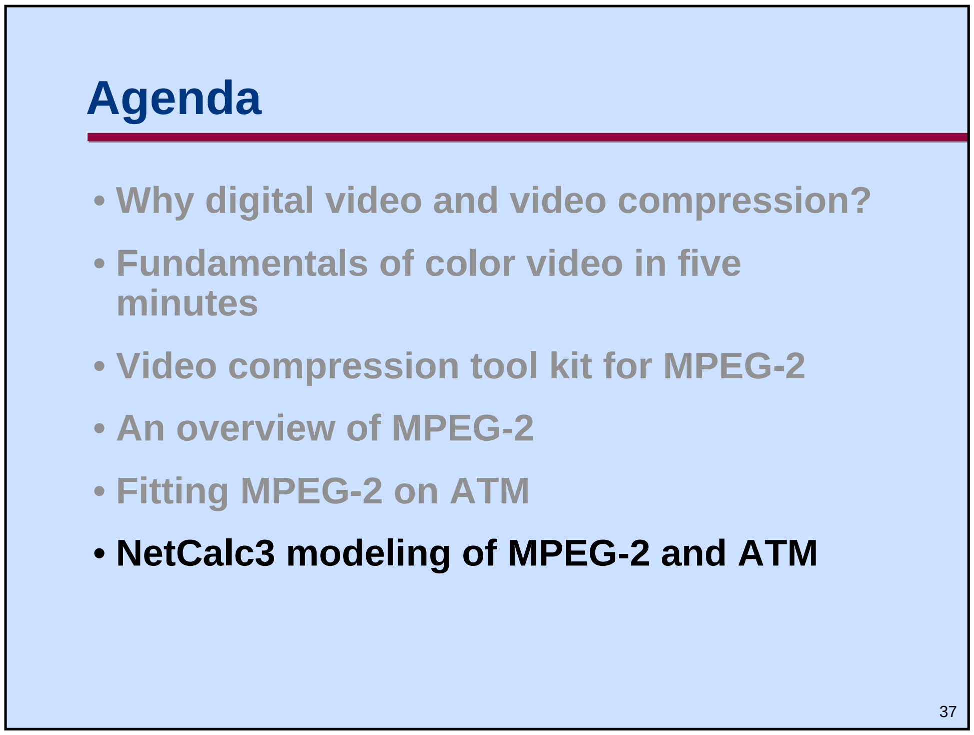
Where to do Adaptation to ATM?

- At the network interface
  - this implies MPEG-2 packet format transmission to the network
    - not a problem with private networks
    - for public networks, requires access rates in an inconvenient speed range (4 Mbit/s to 45 Mbit/s)
- Using customer equipment
  - requires a local MPEG-2/ATM access device
  - network access is ATM UNI
- Using mixture, for example:
  - a local MPEG-2/ATM access device for coder end
  - 4 Mbit/s to 9 Mbit/s access at decoder end

Network traffic calculations are the same for all above cases

36

Adaptation to ATM can be done several different ways or in different places as noted above, and the decision on how to do it can have important ramifications on network cost and viability—but has little, if any, impact on traffic calculations. So for network topology design, there is also minimal impact.

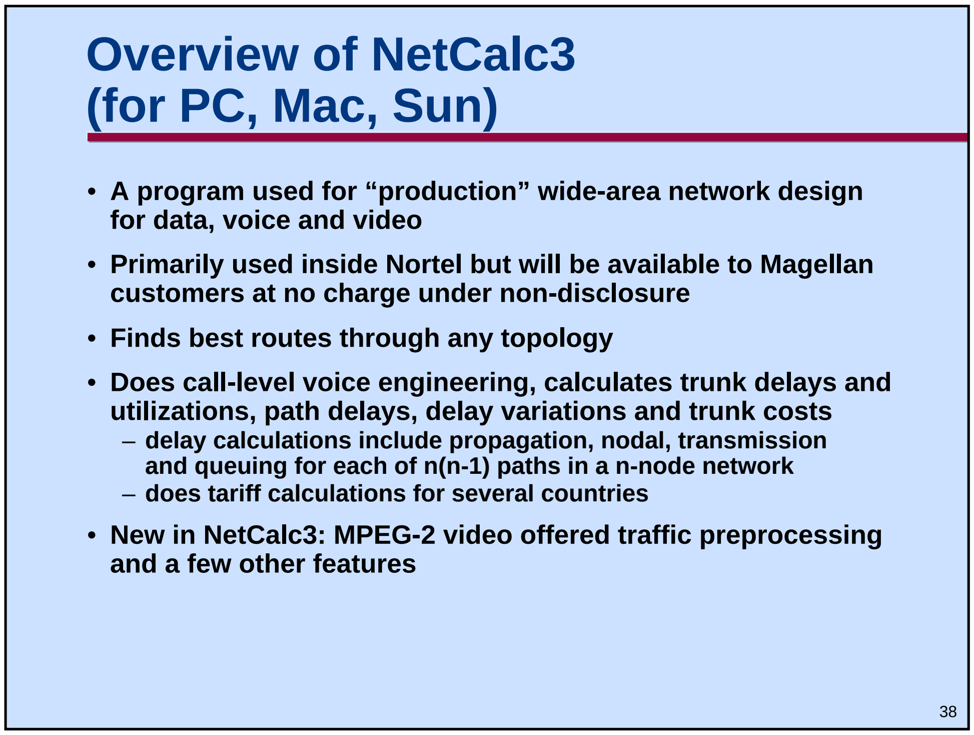


Agenda

- Why digital video and video compression?
- Fundamentals of color video in five minutes
- Video compression tool kit for MPEG-2
- An overview of MPEG-2
- Fitting MPEG-2 on ATM
- **NetCalc3 modeling of MPEG-2 and ATM**

37

A description of the NetCalc3 modeling program and the modeling of MPEG-2 on ATM follow next.



Overview of NetCalc3 (for PC, Mac, Sun)

- **A program used for "production" wide-area network design for data, voice and video**
- **Primarily used inside Nortel but will be available to Magellan customers at no charge under non-disclosure**
- **Finds best routes through any topology**
- **Does call-level voice engineering, calculates trunk delays and utilizations, path delays, delay variations and trunk costs**
  - **delay calculations include propagation, nodal, transmission and queuing for each of n(n-1) paths in a n-node network**
  - **does tariff calculations for several countries**
- **New in NetCalc3: MPEG-2 video offered traffic preprocessing and a few other features**

38

NetCalc was designed to improve productivity and accuracy in repetitive network topology designs, a process sometimes called "production" network engineering. NetCalc can be used for TDM, frame-based networks with pre-emptive queues and ATM. Traffic may include any mixture of data, voice and video.

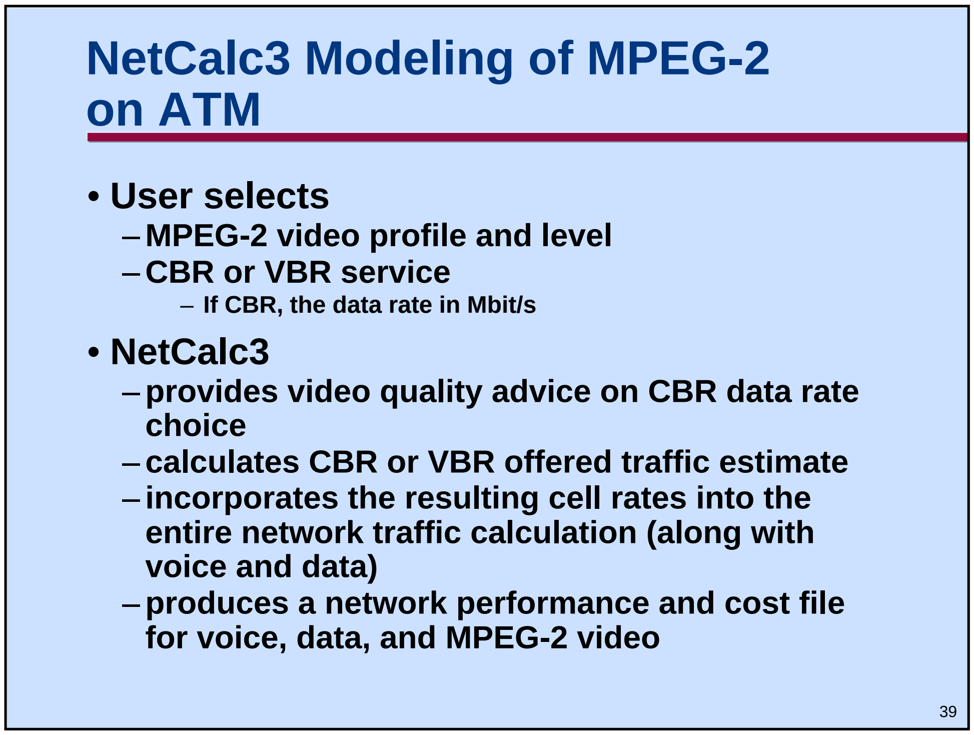
NetCalc was designed primarily for internal use but has been given to some Magellan customers under non-disclosure agreements. Nortel usually works with the customer in doing an initial design or network analysis, and the customer then continues the process on its own. No program, including NetCalc3, can replace a competent network designer; they just make the job easier, faster and more accurate under the direction of a knowledgeable user. Network design beyond a few nodes involves a surprising amount of processing and database searching and extraction, and, because there is usually a tight time frame to be met, a powerful computer aid is mandatory.

NetCalc3 can automatically find the best routes through any topology in which the maximum number of hops between any source and destination node is 12 or fewer.

- A hop is defined as a link or trunk between two nodes.
- The "best" path usually means the most direct way from A to B, and therefore is a route with fewest hops—with ties resolved by shortest distance if there is more than one route with fewest hops. Different routing algorithms can be used in NetCalc3, and the user can override any route chosen by the program.

NetCalc is a delay-based tool, and has an expanding global tariff capability.

The latest version (NetCalc3) is the capability to do MPEG-2 video traffic preprocessing—a task increasingly necessary for the design of ATM networks.



NetCalc3 Modeling of MPEG-2 on ATM

- User selects
  - MPEG-2 video profile and level
  - CBR or VBR service
    - If CBR, the data rate in Mbit/s
- NetCalc3
  - provides video quality advice on CBR data rate choice
  - calculates CBR or VBR offered traffic estimate
  - incorporates the resulting cell rates into the entire network traffic calculation (along with voice and data)
  - produces a network performance and cost file for voice, data, and MPEG-2 video

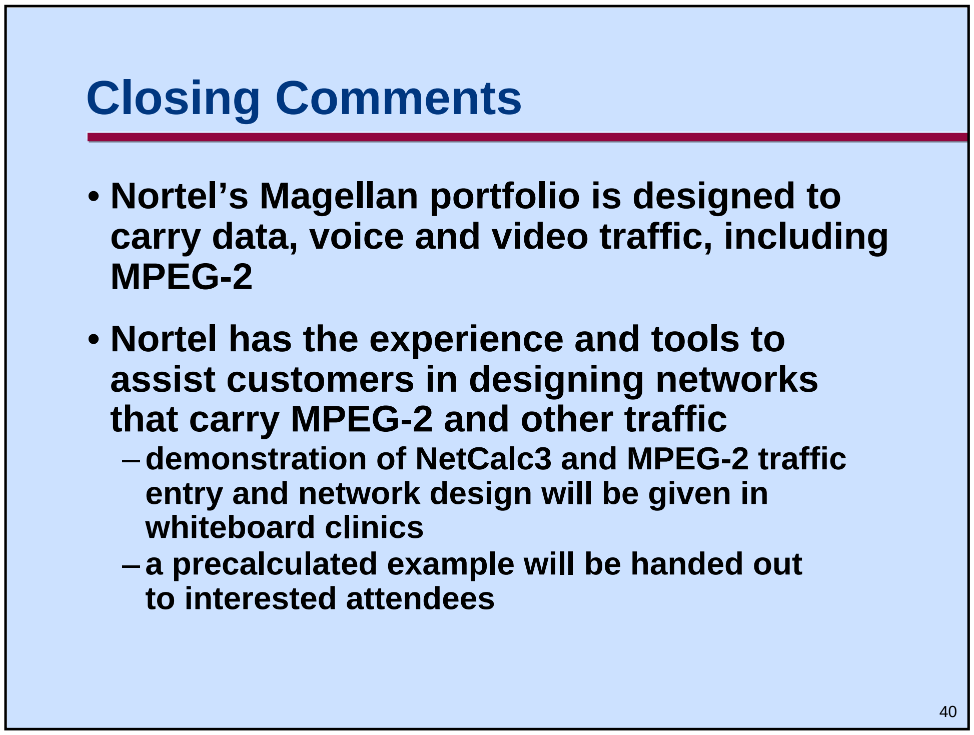
39

A demonstration of NetCalc3 will be given at one or more whiteboard clinics. Other information on the process of design will be provided there as well. This is a brief summary of what the user has to do for MPEG-2 on ATM:

- The user has to know about MPEG-2 video profiles and levels but does not have to know how to translate that into offered traffic or resulting trunk traffic. So the user selects profile and level (i.e. main/main) for any path where there is to be video traffic (i.e. Tokyo to Osaka).
- The user selects service type and speed for the path (i.e. CBR and 7 Mbit/s)

NetCalc3

- provides feedback, if necessary for validity, on the selections;
- calculates the MPEG-2 offered traffic to the network;
- does autorouting;
- incorporates the resulting cell rates into the entire network traffic calculation; and
- provides a complete network-level performance file for data, voice and video:
  - utilizations and delays for all ATM trunks
  - delays for all paths through the network for CBR and VBR traffic
  - cost of all facilities
  - (numerous other outputs available in the whiteboard session demonstrations)



Closing Comments

- Nortel's Magellan portfolio is designed to carry data, voice and video traffic, including MPEG-2
- Nortel has the experience and tools to assist customers in designing networks that carry MPEG-2 and other traffic
  - demonstration of NetCalc3 and MPEG-2 traffic entry and network design will be given in whiteboard clinics
  - a precalculated example will be handed out to interested attendees

40

This workshop is largely product-independent but is relevant to Nortel's Magellan product line application because the Magellan ATM product family is designed to carry data, voice and video—including MPEG-2 coded video.

Nortel has the experience and tools to assist customers in designing multimedia ATM networks. The latest version of one of those tools makes the addition of MPEG-2 traffic to such networks relatively easy compared to earlier approaches.

We invite you to attend the whiteboard sessions for further information on the subject.

Finally, the last pages of the presentation notes shows

- A list of useful references for MPEG-2 on ATM that was used as a source for some of the material in the presentation.
- A list of common terms used for MPEG-2 and ATM.

Appendix A: References

[1] International Standard ISO/IEC 13818 (four parts plus annexes)

[2] "Digital Television" by Dimitris Anastassiou, Proceedings of the IEEE, Vol. 82, No. 4, April 1994

[3] "The Grand Alliance HDTV System" by E. Petajan and J. Mailhot, SPIE Vol. 2186 Image and Video Compression (1994)

[4] "Video Compression for the Grand Alliance: A Historical Perspective" by X. Lebegue et al, International Journal of Imaging Systems and Technology, Vol. 5, 253-258 (1994)

[5] "MPEG-2 Over ATM for Video Dial Tone Networks: Issues and Strategies", by S. Dixit and P. Skelly, IEEE Network, Sept/Oct 1995

{6} "MPEG-2 Systems" by A.G. MacInnis, SPIE Vol. 2187 (1994)

[7] "Performance Evaluation of Nonscalable MPEG-2 Video Coding" by R.L. Schmidt et al, SPIE Vol. 2308 (1994)

[8] "What's Behind DBS Services: MMIC Technology and MPEG Digital Video Compression" by P.M. Bacon and E. Filtzer, IEEE Transactions on Microwave Theory and Techniques, Vol.43, No. 7, July 1995

[9] "MPEG-4: Very low bit rate coding for multimedia applications" by I. Corset et al, SPIE Vol. 2308 (1994)

[10] "Digital Video Coding Techniques for US High-Definition TV" by E. Petajan, IEEE Micro (magazine), Oct 1992

[11] "A 160-Mpixel/s IDCT Processor for HDTV" by P.A. Ruetz and P. Tong, IEEE Micro (magazine), Oct 1992

[12] "Digital Visual Communications over Telephone Networks" by C.T. Chen et al, Journal of Visual Communication and Image Representation, Academic Press, Inc., June 1995

[13] "Source Models for VBR Broadcast-Video Traffic" by D.P. Heyman and T.V. Lakshman, IEEE/ACM Transactions on Networking, Feb 1996

[14] "Cell Multiplexing in ATM Networks" by Z. Rosberg, IEEE/ACM Transactions on Networking, Feb 1996

Appendix B: MPEG-2 and ATM Terminology

AAL	ATM adaptation layer, which does the SAR function (defined below) and ensures specified QoS for the applications
AAL-1	Defined for circuit emulation (real-time CBR) across an ATM network (and readily usable for MPEG-2 video)
AAL-2	Intended to provide real-time services for connection-oriented VBR applications (not defined as of early 1996, and not yet usable for MPEG-2)
AAL-3/4	Defined for connectionless (datagram) VBR services over pre-established connections (not designed for connection-oriented applications such as MPEG-2)
AAL-5	Defined for connection-oriented VBR data services (usable for MPEG-2, even though MPEG-2 video is a real-time application)
ATM	Asynchronous transfer mode
CBR	Constant bit rate (service on ATM)
CCIR 601	An accurate digital version of NTSC, PAL, SECAM (used in MPEG-2)
CIF	Common intermediate format (video)
CIR	Committed information rate (frame relay)
DCT	Discrete cosine transform
FDCT	Forward discrete cosine transform
GOP	Group of pictures, for example IBBPBBPBB
HDTV	High definition television
IDCT	Inverse discrete cosine transform
JPEG	Joint Photographic Expert Group
LOSSY	Refers to compression techniques in which the source is not perfectly reproducible although the reproduction quality may be good to excellent
Pel	Short for pixel
PCR	Program clock references, or time stamps in the TS of MPEG-2
PES	Packetized elementary stream in MPEG-1 and 2
PS	Program Stream of MPEG-1 and 2
Pixel	Picture element, or smallest possible dot on a video monitor
QoS	Quality of service, definition of which varies by service
RGB	Red, green, blue color signal format
SAR	Segmentation (of frames to cells) and reassembly (of the frames from cells)
SCR	System clock references, or time stamps in the PS of MPEG-1 or 2
SIF	Standard intermediate format (video)
TP	The 188-byte transport packets in TS
TS	Transport Stream of MPEG-2
UNI	User to network interface
VBR	Variable bit rate traffic or service on ATM, connection;ess or connection-oriented