

.....

18-796



Multimedia Communications:
Coding, Systems, and Networking

Prof. Tsuhan Chen
tsuhan@ece.cmu.edu

.....

MPEG-1 Video



•
•
•

Moving Picture Expert Group

- Moving Pictures Expert Group (MPEG)
 - ISO/IEC JTC1/SC29/WG11
 - Formed in Jan. 1988
- MPEG standards
 - MPEG-1 (ISO/IEC 11172, Nov 92)
 - MPEG-2 (ISO/IEC 13818, Nov 94)
 - MPEG-4 (ISO/IEC 14496, Oct 98)
 - MPEG-7 (ongoing)
- Specify only bitstream syntax and decoding

18-796/Spring 1999/Chen

•
•
•

MPEG-1 vs. MPEG-2

- MPEG-1
 - Audio and video on storage media such as CD-ROM
 - 1x CD-ROM drive: 150 KB/s = 1.2 Mbits/s
 - Targeted at 1 to 1.5 Mbits/s
 - ~1.2 Mbits/s for video, ~250 kbits/s for audio
- MPEG-2
 - Digital TV: SDTV, HDTV
 - Wide range of bit rates 4 to 80 Mbits/s
 - Optimized around 4 Mbits/s
 - Supports interlaced video and scalable coding

18-796/Spring 1999/Chen

•
•
•

MPEG-1

- Outline
 - Applications and history
 - Requirements
 - New features (w.r.t. H.261)
 - Simulation Model 3
 - Bitstream syntax
 - Video buffering verifier

18-796/Spring 1999/Chen

•
•
•

MPEG-1

- Applications
 - Video on digital storage media
 - Computer and telecommunication networks
- History
 - 1988: started
 - Sep. 1989: Proposal registration
 - Oct. 1989: Subjective tests
 - Mar. 1990: Definition of video algorithm (Simulation Model 1)
 - Sep. 1990: Committee Draft for video

Competition

Convergence

18-796/Spring 1999/Chen

•
•
•

Parts of MPEG-1

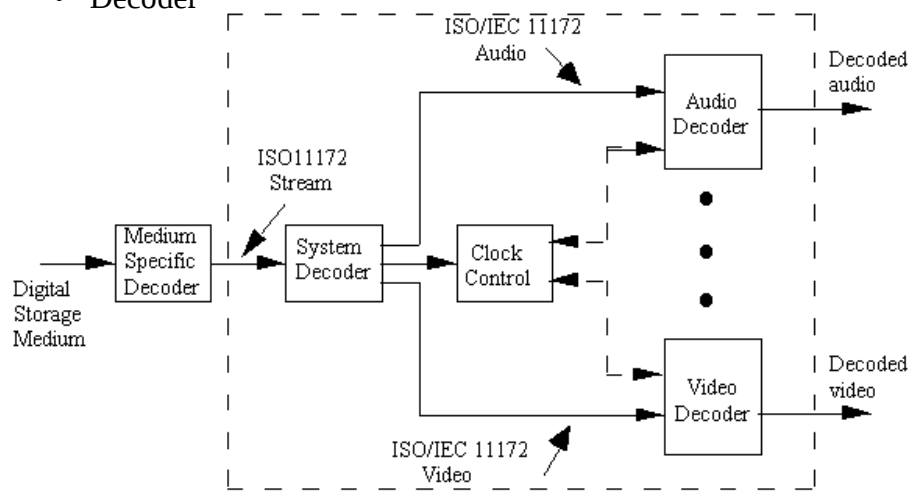
- ISO/IEC 11172-1: Systems
- ISO/IEC 11172-2: Video
- ISO/IEC 11172-3: Audio
- ISO/IEC 11172-4: Conformance Testing
- ISO/IEC 11172-5: Software

18-796/Spring 1999/Chen

•
•
•

MPEG-1 System

- Decoder



18-796/Spring 1999/Chen

•
•
•

Requirements: Basic

- Coding of generic video with good quality (about VHS video) at 1 to 1.5 Mbits/s
- Random access to a frame in limited time
 - Frequent access points
- Fast forward/reverse
 - Seek and play in FF/FR using access points
- System supporting audio-visual synchronized play/access
- A practical/implementable decoder

18-796/Spring 1999/Chen

•
•
•

Requirements: Additional

- Reverse playback
 - Access points and buffering
- Tradeoff quality with coding/decoding delay
 - Delay between 150 ms to 1 sec
- Source coding scheme and bits organization to be robust to errors
- Some flexibility in picture format to support variety of applications
- Real-time encoder at reasonable cost

18-796/Spring 1999/Chen

New Features (w.r.t. H.261)

- Flexible picture sizes, picture rates, etc.
- Flexible slice structure instead of GOB
- Group of pictures (GOP)
- Bi-directional motion compensation
- Half-pel motion compensation
- Separate MB-type VLC tables for I, P, and B pictures
- Quantization tables
- VLC for MV difference
- VLC supports large range of DCT coefficients

18-796/Spring 1999/Chen

Parameters

- Picture size up to 4096×4096 supported
 - Normally at 360×240
- Pel aspect ratio: 14 choices
- Picture rates: 23.976, 24 (movies), 25 (PAL), 29.97, 30, 50, 59.94, 60

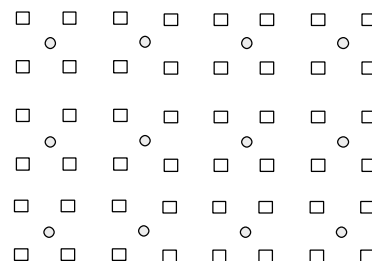
- 4:2:0 format

– (same as H.26x)

□ = Y Pels

○ = C_u and C_v Pels

= Block Edges



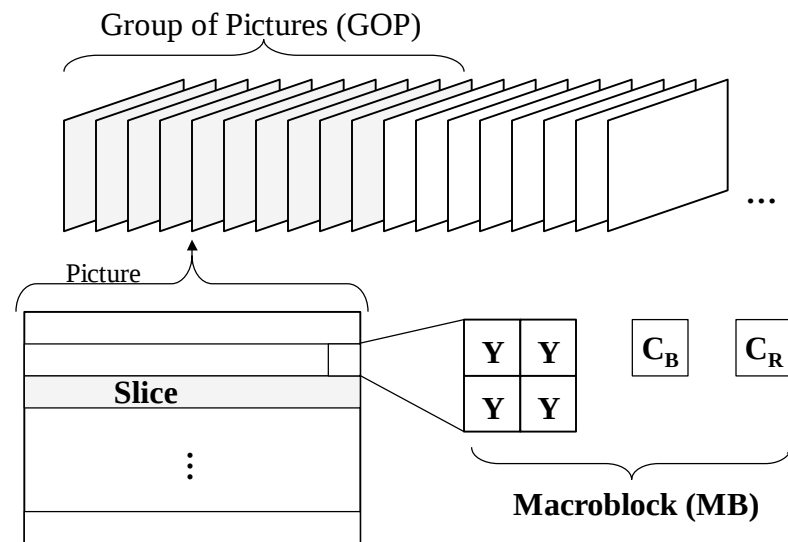
Constrained Parameters

- To allow interoperability of MPEG-1 equipment, a subset of parameter space is defined
- All MPEG-1 conformant decoders shall decode constrained parameter bitstreams

Horizontal picture size	≤ 768 pels
Vertical picture size	≤ 576 lines
Picture area	≤ 396 macroblocks
Pel rate	$\leq 396 \times 25$ macroblocks per second
Picture rate	≤ 30 Hz
Motion vector range	< -64 to $+63.5$ pels (using half-pel vectors), etc.
Input buffer size (in VBV model)	$\leq$ to 327 680 bits
Bitrate	$\leq 1\ 856\ 000$ bits/second (constant bitrate)

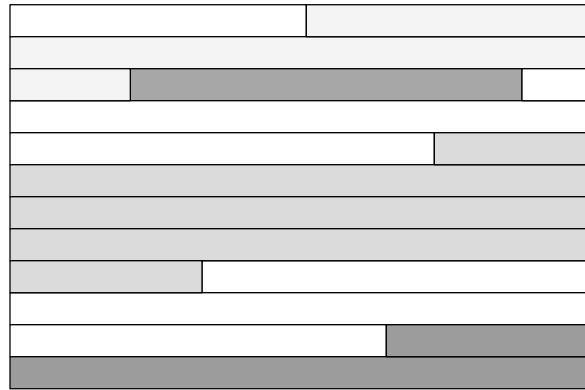
18-796/Spring 1999/Chen

Data Structure



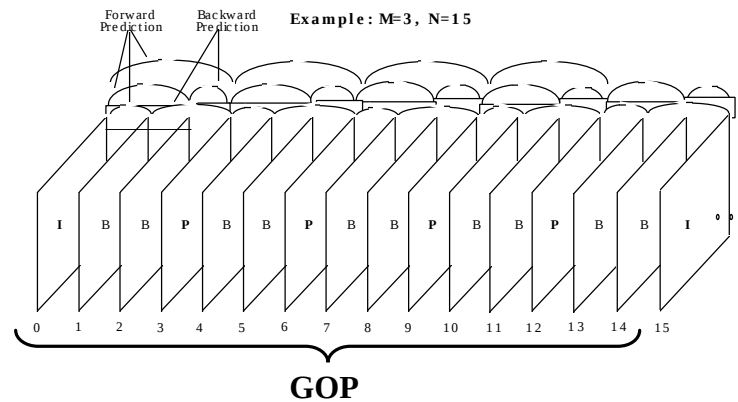
18-796/Spring 1999/Chen

Possible Slice Arrangement



18-796/Spring 1999/Chen

Group of Picture (GOP)

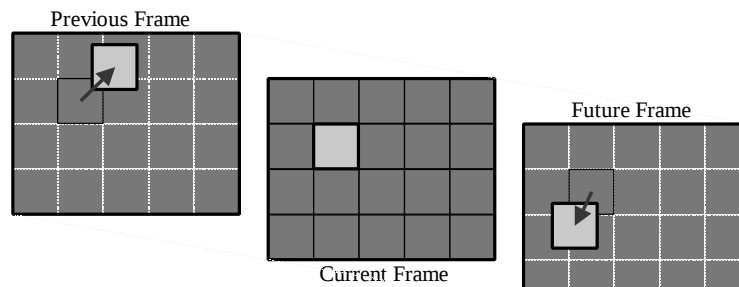


- N = number of pictures in a GOP
- M = prediction distance ($M-1$ in-between B-pictures)
- Tradeoff on M

18-796/Spring 1999/Chen

Bi-Directional Prediction

- Prediction from the previous frame, or the prediction from the future frame, or an average of both is used as the final prediction.
- The prediction error is then coded and transmitted



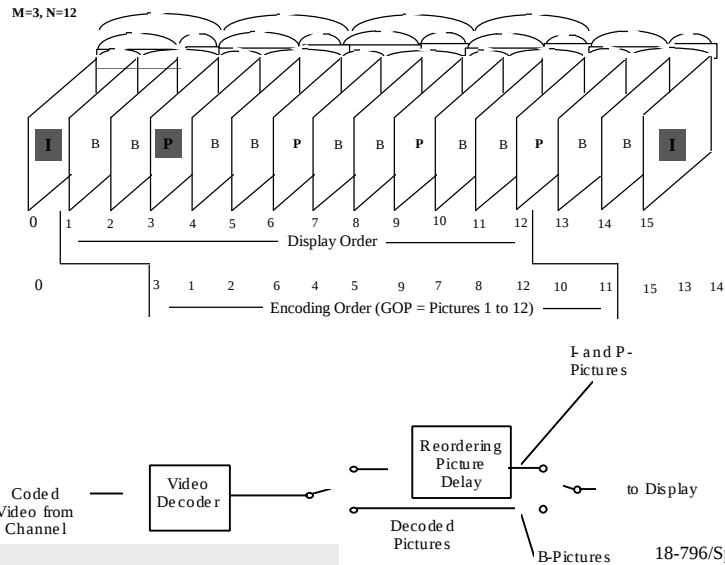
18-796/Spring 1999/Chen

Bi-Directional Prediction (cont.)

- Advantages
 - Higher coding efficiency
 - Increased frame rate with few extra bits
 - No uncovered background problem
 - No error propagation
- Disadvantages
 - More frame storage is needed
 - More delay

18-796/Spring 1999/Chen

Delay



18-796/Spring 1999/Chen

Motion Compensation

- Same as H.263
 - Half-pel resolution for motion vectors
 - Bilinear interpolation
 - Differential coding of motion vectors
 - Motion compensation on 16×16 luminance blocks
 - Motion vectors divided by 2 for chrominance
- Different from H.263
 - Separate MB-type VLC tables for I, P, and B pictures
 - VLC for MV difference

18-796/Spring 1999/Chen

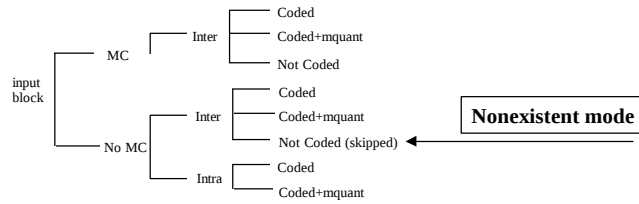
MB Type

- For I-pictures

mb_type modes	mb_quant	mb_mot forward	mb_mot backward	mb_ pattern	mb_intra	VLC
Intra					y	1
Intra+mquant	y				y	01

- For P-pictures

- MB Type



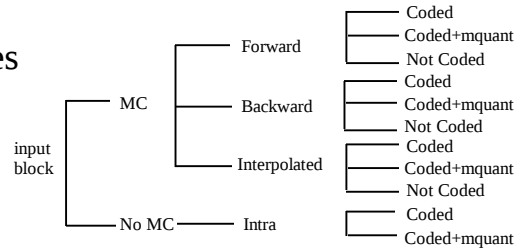
- VLC Table

mb_type modes	mb_quant	mb_mot forward	mb_mot backward	mb_ pattern	mb_intra	VLC
MC, Coded		y		y		1
No MC, Coded				y		01
MC, Not Coded		y				001
Intra					y	00011
MC, Coded+mquant	y	y		y		00010
No MC, Coded+mquant	y			y		00001
Intra+mquant	y				y	000001

MB Type (cont.)

- For B-pictures

- MB Types



- VLC Table

mb_type modes	mb_quant	mb_mot forward	mb_mot backward	mb_ pattern	mb_intra	VLC
MC Interpolated, Not Coded		y	y			10
MC Interpolated, Coded		y	y	y		11
MC Backward, Not Coded			y			010
MC Backward, Coded			y	y		011
MC Forward, Not Coded		y				0010
MC Forward, Coded		y		y		0011
Intra					y	00011
MC Interpol., Coded+mquant	y	y	y	y		00010
MC Forward, Coded+mquant	y	y		y		000011
MC Backward, Coded+mquant	y		y	y		000010
Intra+mquant	y				y	000001

VLC for MV Difference

$f=1,2,4,8,16,32,64$

$dMD = MD - PMD$

$motion_code = [dMD + Sign(dMD) \times (f-1)] / f$



$r = |motion_code \times f| - |dMD|$

The residual r is sent with $\log(f)$ bits

$dMD = motion_code \times f$

$- Sign(motion_code) \times r$

motion_code	VLC
0	1
1	01s
2	001s
3	0001 s
4	0000 11s
5	0000 101s
6	0000 100s
7	0000 011s
8	0000 0101 1s
9	0000 0101 0s
10	0000 0100 1s
11	0000 0100 01s
12	0000 0100 00s
13	0000 0011 11s
14	0000 0011 10s
15	0000 0011 01s
16	0000 0011 00s

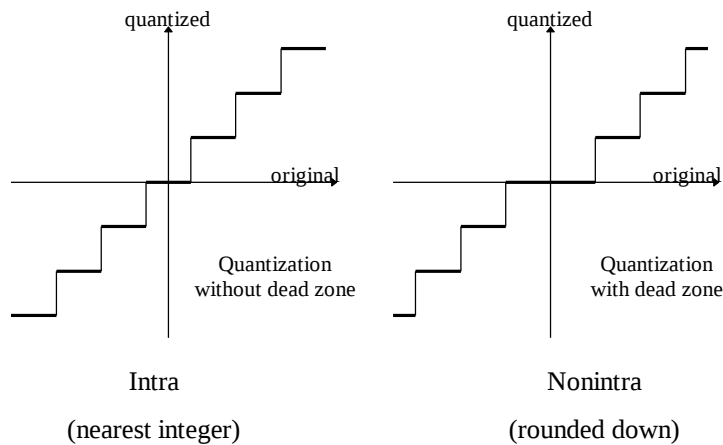
– Range $[-512, +511.5]$ for half-pel motion vectors

– Range $[-1024, +1023]$ for full-pel motion vectors

18-796/Spring 1999/Chen

Quantization

- Same as H.263



18-796/Spring 1999/Chen

Quantization Tables

- Default quantization tables

8	16	19	22	26	27	29	34
16	16	22	24	27	29	34	37
19	22	26	27	29	34	34	38
22	22	26	27	29	34	37	40
22	26	27	29	32	35	40	48
26	27	29	32	35	40	48	58
26	27	29	34	38	46	56	69
27	29	35	38	46	56	69	83

intra

16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16

inter

18-796/Spring 1999/Chen

Coding for Intra DC

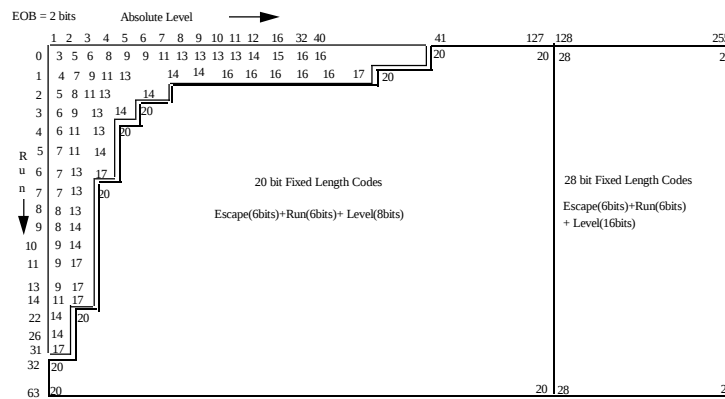
- VLC and VLI for Intra DC differential values

Differential DC	SIZE	VLC		VLI
		Luminance	Chrominance	
-255 to -128	8	1111 110	1111 1110	00000000 to 01111111
-127 to -64	7	1111 10	1111 110	0000000 to 0111111
-63 to -32	6	1111 0	1111 10	000000 to 011111
-31 to -16	5	1110	1111 0	00000 to 01111
-15 to -8	4	110	1110	0000 to 0111
-7 to -4	3	101	110	000 to 011
-3 to -2	2	01	10	00 to 01
-1	1	00	01	0
0	0	100	00	
1	1	00	01	1
2 to 3	2	01	10	10 to 11
4 to 7	3	101	110	100 to 111
8 to 15	4	110	1110	1000 to 1111
16 to 31	5	1110	1111 0	10000 to 11111
32 to 63	6	1111 0	1111 10	100000 to 111111
64 to 127	7	1111 10	1111 110	1000000 to 1111111
128 to 255	8	1111 110	1111 1110	10000000 to 11111111

18-796/Spring 1999/Chen

2D VLC

- Similar to H.261
 - Most codes same, some codes shorter
 - Levels up to 255



18-796/Spring 1999/Chen

2D VLC (cont.)

Run	Level	Code Word	Run	Level	Code Word
0	1	1s (if first inter-coefficient)	4	1	0011 0s
0	2	11s (otherwise)	4	2	0000 0011 11s
0	3	0100 s	4	3	0000 0001 0010s
0	4	0010 1s	5	1	0001 11s
0	5	0000 110s	5	2	0000 0010 01s
0	6	0010 0110s	5	3	0000 0000 1001 0s
0	7	0000 0001s	6	1	0001 01s
0	8	0000 0001 10s	6	2	0000 0001 1110s
0	9	0000 0001 1000s	7	1	0001 00s
0	10	0000 0001 0011s	7	2	0000 0001 0101s
0	11	0000 0001 0000s	8	1	0000 111s
0	12	0000 0000 1101s	8	2	0000 0001 0001s
0	13	0000 0000 1100s	9	1	0000 101s
0	14	0000 0000 1010s	9	2	0000 1000 1s
0	15	0000 0000 1011s	10	1	0010 0111 s
1	1	0000 0000 1011s	10	2	0000 0000 1000 0s
1	2	0001 10s	11	1	0010 0011 s
1	3	0010 0101s	12	1	0010 0010 s
1	4	0000 0001 00s	13	1	0010 0000 s
1	5	0000 0001 1011s	14	1	0000 0011 10s
1	6	0000 0000 1011 0s	15	1	0000 0010 01s
1	7	0000 0000 1010 1s	16	1	0000 0010 00s
1	8	0000 0000 1010 1s	17	1	0000 0000 1111s
2	1	0101 s	18	1	0000 0001 1010s
2	2	0000 100s	19	1	0000 0001 1001s
2	3	0000 0010 11s	20	1	0000 0001 0111s
2	4	0000 0001 0100s	21	1	0000 0001 0110s
2	5	0000 0000 1010 0s	22	1	0000 0000 1111 1s
3	1	0011 1s	23	1	0000 0000 1111 0s
3	2	0010 0100s	24	1	0000 0000 1110 1s
3	3	0000 0001 1100s	25	1	0000 0000 1110 0s
3	4	0000 0001 1001 1s	26	1	0000 0000 1101 1s
EOB	10	0000 01			
Escape	0000 01				

18-796/Spring 1999/Chen

2D VLC (cont.)

Run Level	Code Word	Run Level	Code Word
0	8*0 0111 11s	1	8*0 0011 111s
16	8*0 0111 10s	8	8*0 0011 110s
17	8*0 0111 10s	9	8*0 0011 101s
18	8*0 0111 01s	10	8*0 0011 101s
19	8*0 0111 00s	11	8*0 0011 100s
20	8*0 0110 11s	12	8*0 0011 011s
21	8*0 0110 10s	13	8*0 0011 010s
22	8*0 0110 01s	14	8*0 0011 001s
23	8*0 0110 00s	15	8*0 0001 0011s
24	8*0 0101 11s	16	8*0 0001 0010s
25	8*0 0101 10s	17	8*0 0001 0001s
26	8*0 0101 01s	18	8*0 0001 0000s
27	8*0 0101 00s	6	8*0 0001 0100s
28	8*0 0100 11s	11	8*0 0001 1010s
29	8*0 0100 10s	12	8*0 0001 1001s
30	8*0 0100 01s	13	8*0 0001 1000s
31	8*0 0100 00s	14	8*0 0001 0111s
32	8*0 0011 000s	15	8*0 0001 0110s
33	8*0 0010 111s	16	8*0 0001 0101s
34	8*0 0010 110s	27	8*0 0001 1111s
35	8*0 0010 101s	28	8*0 0001 1110s
36	8*0 0010 100s	29	8*0 0001 1101s
37	8*0 0010 011s	30	8*0 0001 1100s
38	8*0 0010 010s	31	8*0 0001 1011s
39	8*0 0010 001s		
40	8*0 0010 000s		

8*0 = 0000 0000

18-796/Spring 1999/Chen

2D VLC (cont.)

fixed length code	run
0000 00	0
0000 01	1
0000 10	2
...	...
...	...
...	...
...	...
...	...
...	...
1111 11	63

fixed length code	level
forbidden	-256
1000 0000 0000 0001	-255
1000 0000 0000 0010	-254
...	...
1000 0000 0111 1111	-129
1000 0000 1000 0000	-128
1000 0001	-127
1000 0010	-126
...	...
1111 1110	-2
1111 1111	-1
forbidden	0
0000 0001	1
...	...
0111 1111	127
0000 0000 1000 0000	128
0000 0000 1000 0001	129
...	...
0000 0000 1111 1111	255

18-796/Spring 1999/Chen

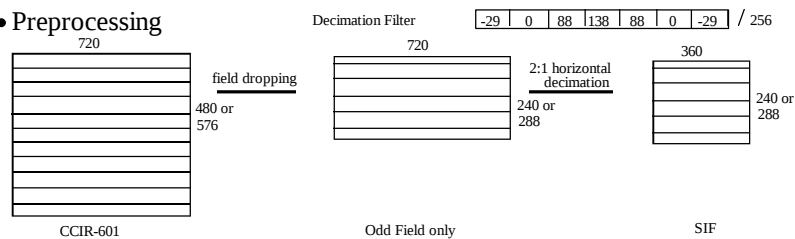
Simulation Model (SM) 3

- Pre- and Post- processing
- Motion estimation/compensation
 - Telescopic search
 - Half-pel update
- Mode decision: MC/no MC inter/intra
- Quantization adaptation and rate control
- See Annex D of IS 11172-2

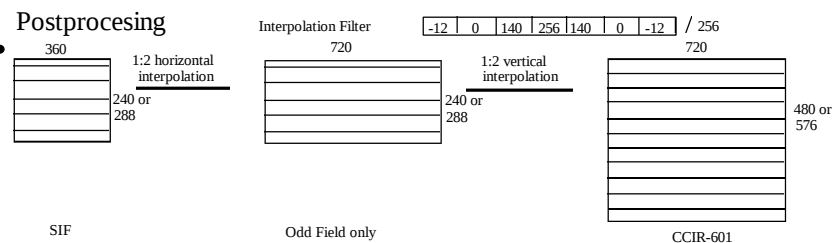
18-796/Spring 1999/Chen

Pre- and Post- Processing

• Preprocessing

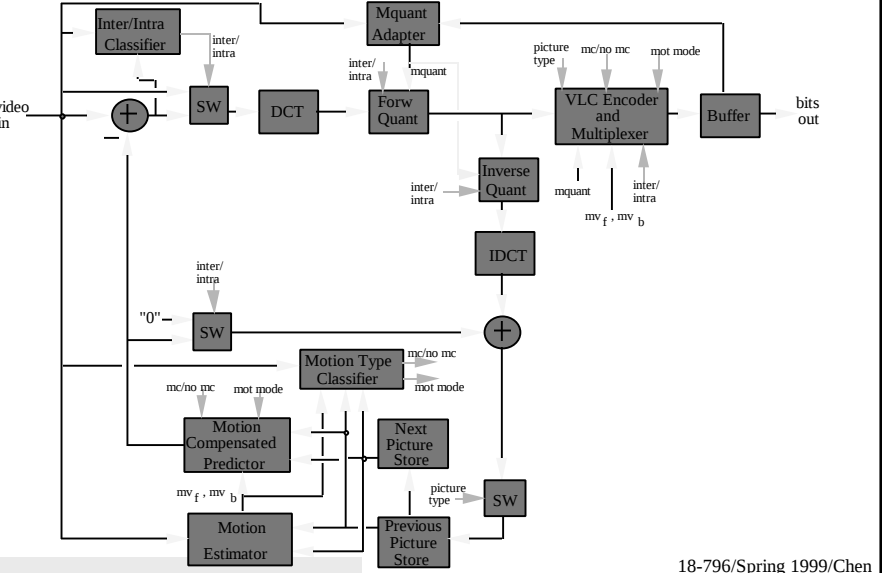


• Postprocessing



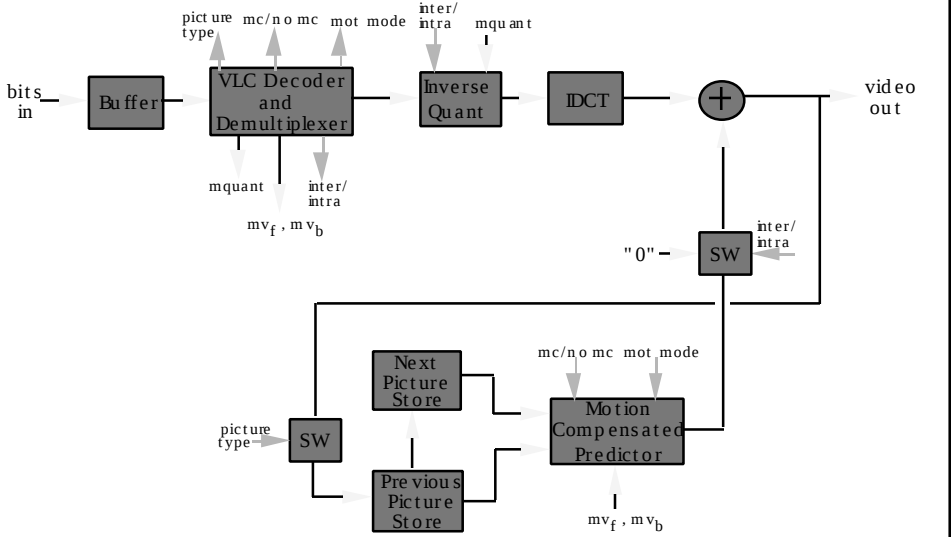
18-796/Spring 1999/Chen

SM 3 Encoder



18-796/Spring 1999/Chen

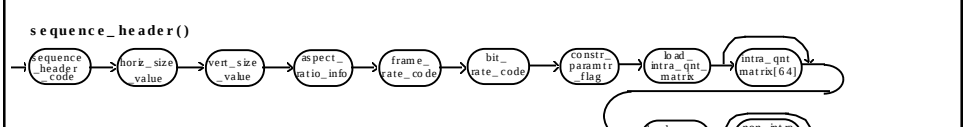
Decoder



18-796/Spring 1999/Chen

- ⋮
- ⋮
- ⋮
- Bitstream Syntax

The diagram illustrates the organization of a bitstream. It consists of a sequence of fields: `sequence_header()`, `user_data()`, `goto_start_of_picture()`, `user_data()`, `picture_header()`, `user_data()`, and `picture_end`. Arrows indicate the flow and dependencies between these fields. The `sequence_header()` field is followed by `user_data()`, which is then followed by `goto_start_of_picture()`. This is followed by another `user_data()` field, then `picture_header()`, then another `user_data()` field, and finally `picture_end`. Arrows show that the `sequence_header()` field is followed by `user_data()`, which is then followed by `goto_start_of_picture()`. This is followed by another `user_data()` field, then `picture_header()`, then another `user_data()` field, and finally `picture_end`. Arrows also show that the `sequence_header()` field is followed by `user_data()`, which is then followed by `goto_start_of_picture()`. This is followed by another `user_data()` field, then `picture_header()`, then another `user_data()` field, and finally `picture_end`.



```
sequence_header()
```

```

sequence_header_code --> hork_size_value --> vert_size_value --> aspect_ratio_info --> frame_rate_code --> bit_rate_code --> constr_paramtr_flag --> load_intra_qnt_mattrk
load_intra_qnt_mattrk --> intra_qnt_mattrk[64]
load_intra_qnt_mattrk --> load_inter_qnt_mattrk

```



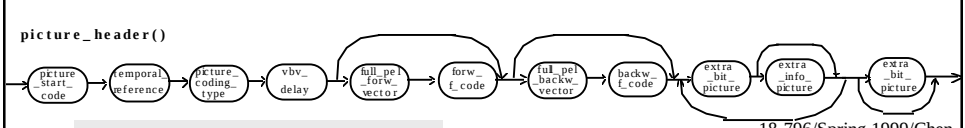
```

graph LR
    user_data_in(( )) --> user_data((user_data  
stan_code))
    user_data --> user_data
    user_data --> intra_qml_matrix[intra_qml  
matrix]
    intra_qml_matrix --> vbo_matrix64[vbo  
matrix 64]
    vbo_matrix64 --> out(( ))

```



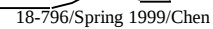
```
graph LR
    group_of_picture_header() --> group_start_code
    group_start_code --> time_code
    time_code --> closed_flag
    closed_flag --> broken_link_flag
    broken_link_flag --> next[ ]
    style next fill:none,stroke:none
```



```

graph LR
    Start(( )) --> picture_start_code[picture_start_code]
    picture_start_code --> temporal_inference[temporal_inference]
    temporal_inference --> picture_coding_type[picture_coding_type]
    picture_coding_type --> vbv_delay[vbv_delay]
    vbv_delay --> full_pel_forw_vector[full_pel_forw_vector]
    full_pel_forw_vector --> forw_f_code[forw_f_code]
    forw_f_code --> full_pel_backw_vector[full_pel_backw_vector]
    full_pel_backw_vector --> backw_f_code[backw_f_code]
    backw_f_code --> extra_bit_picture[extra_bit_picture]
    extra_bit_picture --> extra_info_picture[extra_info_picture]
    extra_info_picture --> extra_bit_picture
    extra_bit_picture --> End(( ))
    style Start fill:none,stroke:none
    style End fill:none,stroke:none

```



- ⋮
- ⋮
- ⋮

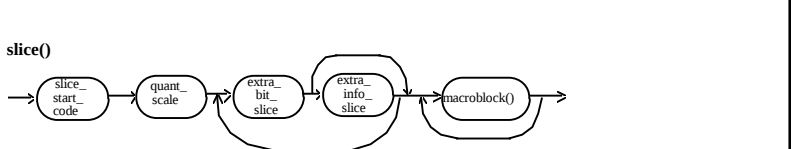
Bitstream Syntax (cont.)

picture_data()

```

graph LR
    Input(( )) --> slice(slice())
    slice --> Output(( ))
    slice --> slice
  
```

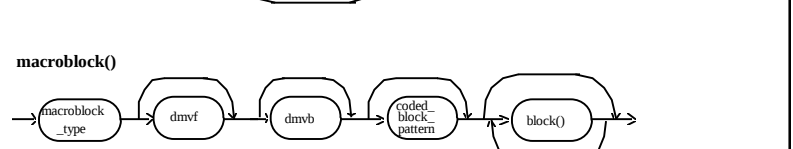
The diagram shows a function call `picture_data()`. Inside the function, there is a loop. An input arrow enters a box labeled `slice()`. From the box, an arrow exits to the right, and another arrow loops back to the input of the `slice()` box.



```

graph LR
    Input(( )) --> slice_start_code[slice_start_code]
    slice_start_code --> quant_scale[quant_scale]
    quant_scale --> extra_bit_slice[extra_bit_slice]
    extra_bit_slice --> extra_info_slice[extra_info_slice]
    extra_info_slice --> macroblock_macroblock[macroblock()]
    extra_info_slice --> extra_bit_slice
    extra_info_slice --> quant_scale
    macroblock_macroblock --> Output(( ))

```



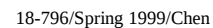
```

graph LR
    macroblock_in(( )) --> macroblock_type[macroblock_type]
    macroblock_type --> dmvf[dmvf]
    dmvf --> dmvb[dmvb]
    dmvb --> coded_block_pattern[coded_block_pattern]
    coded_block_pattern --> block_func[block()]
    block_func --> out(( ))
    macroblock_type --> dmvf
    dmvf --> dmvb
    dmvb --> coded_block_pattern
    coded_block_pattern --> block_func

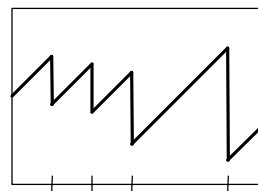
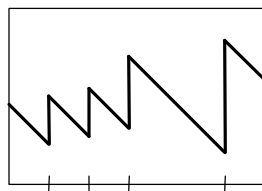
```



```
graph LR
    Input(( )) --> DCT[DCT coefficient]
    DCT --> EndOfBlock([end_of_block])
    EndOfBlock --> Input
```



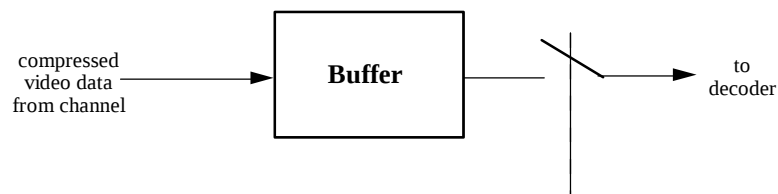
Encoder and Decoder Buffers



18-796/Spring 1999/Chen

Video Buffer Verifier

- Video Buffer Verifier (VBV)
 - Any MPEG-1 bit-stream is prohibited from over/under-flowing the buffer of this VBV
 - For constant bit-rate operation, this puts limitation on the variation of bits per picture



At each picture time, the data for one picture is removed for decoding

18-796/Spring 1999/Chen

•
•
•

Reference

- Joan L. Mitchell et al., *MPEG Video: Compression Standard*, Chapman & Hall, New York, NY