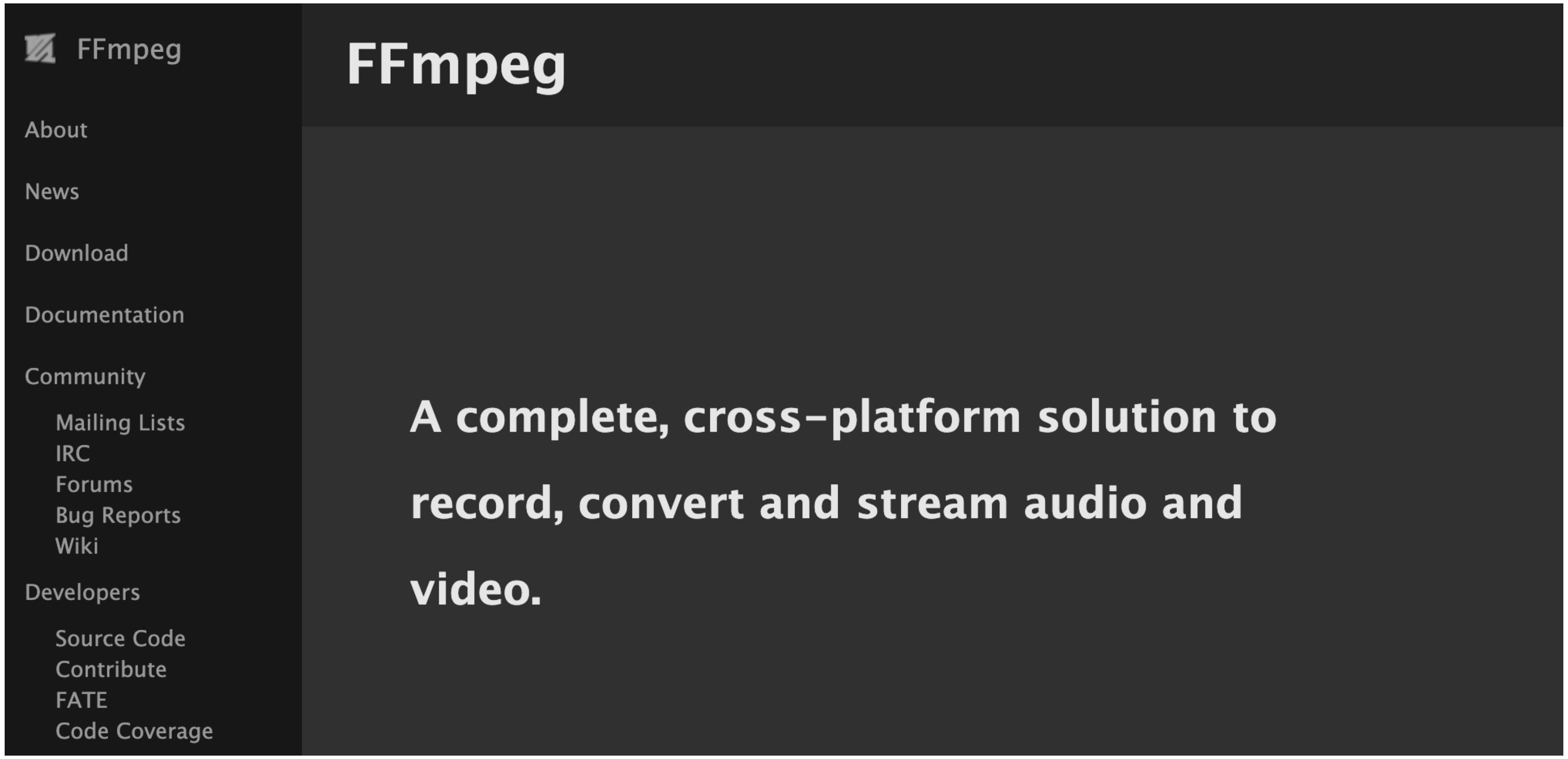
视频技术
Video Technology

matrix-tech-sharing-20190429

THE PEACH OPEN MOVIE PROJECT PRESENTS
Big Buck
BUNNY
WRITTEN AND DIRECTED BY SACHA GOEDEGEBURE – ART DIRECTOR ANDREAS GORALCZYK – LEAD ARTIST ENRICO VALENZA
ANIMATORS NATHAN VEGDAHL, WILLIAM REYNISH – TECHNICAL DIRECTORS CAMPBELL BARTON, BRECHT VAN LOMMEL
MUSIC BY JAN MORGENSTERN – PRODUCED BY TON ROOSENDAAL, BLENDER FOUNDATION
LICENSED AS CREATIVE COMMONS 3.0 ATTRIBUTION
WWW.BIGBUCKBUNNY.ORG
SPONSORS
creative commons nederland
digitale pioniers
wavemage
Maqina
FILMMORE
blenderinstitute

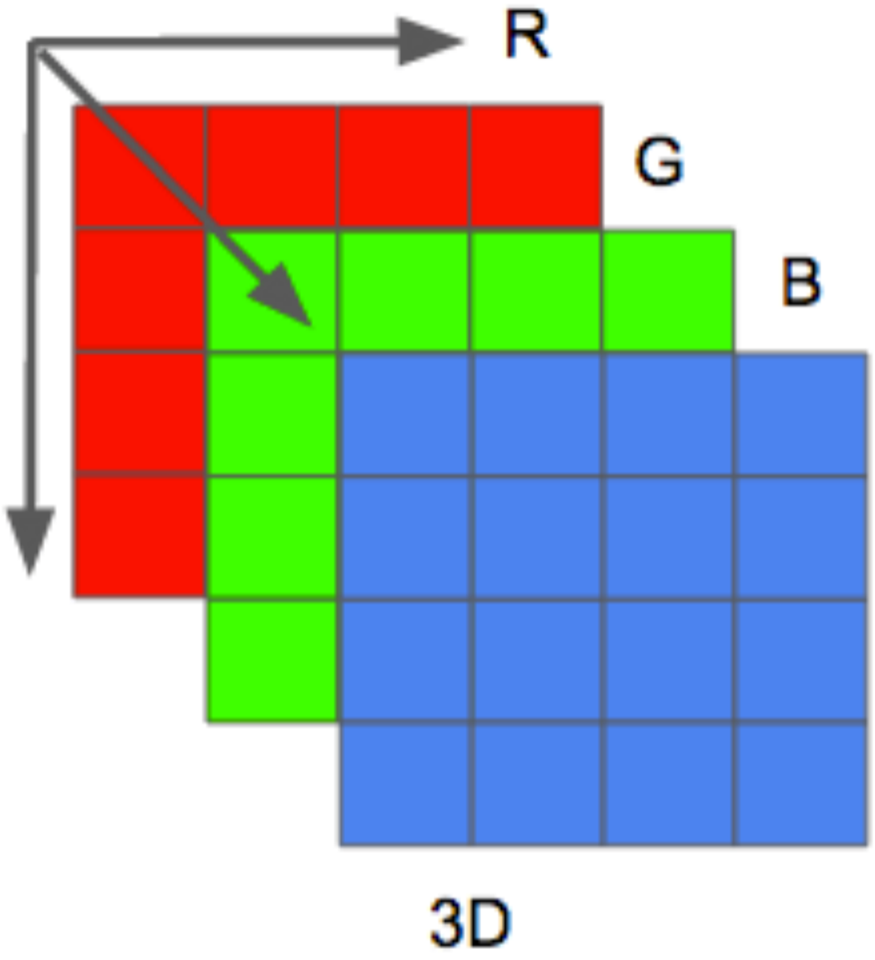
Big Buck Bunny是最有名的开源视频样例, 每个视频开发者都看到吐



FFmpeg

A complete, cross-platform solution to record, convert and stream audio and video.

虽然很常用, 做的人不多. 资料很难找, 全靠啃标准



视频非常简单，就是好多张图片连着放，加上声音

那还这么多人做它干啥呢

一个像素点3个色, 一个色一个Byte (8bits) (24位色)
一个2K普普通通分辨率 1920*1080 的图, 约等于 2MB
一秒钟60帧, 120MB
一个小时的视频 120 * 3600 = 432GB

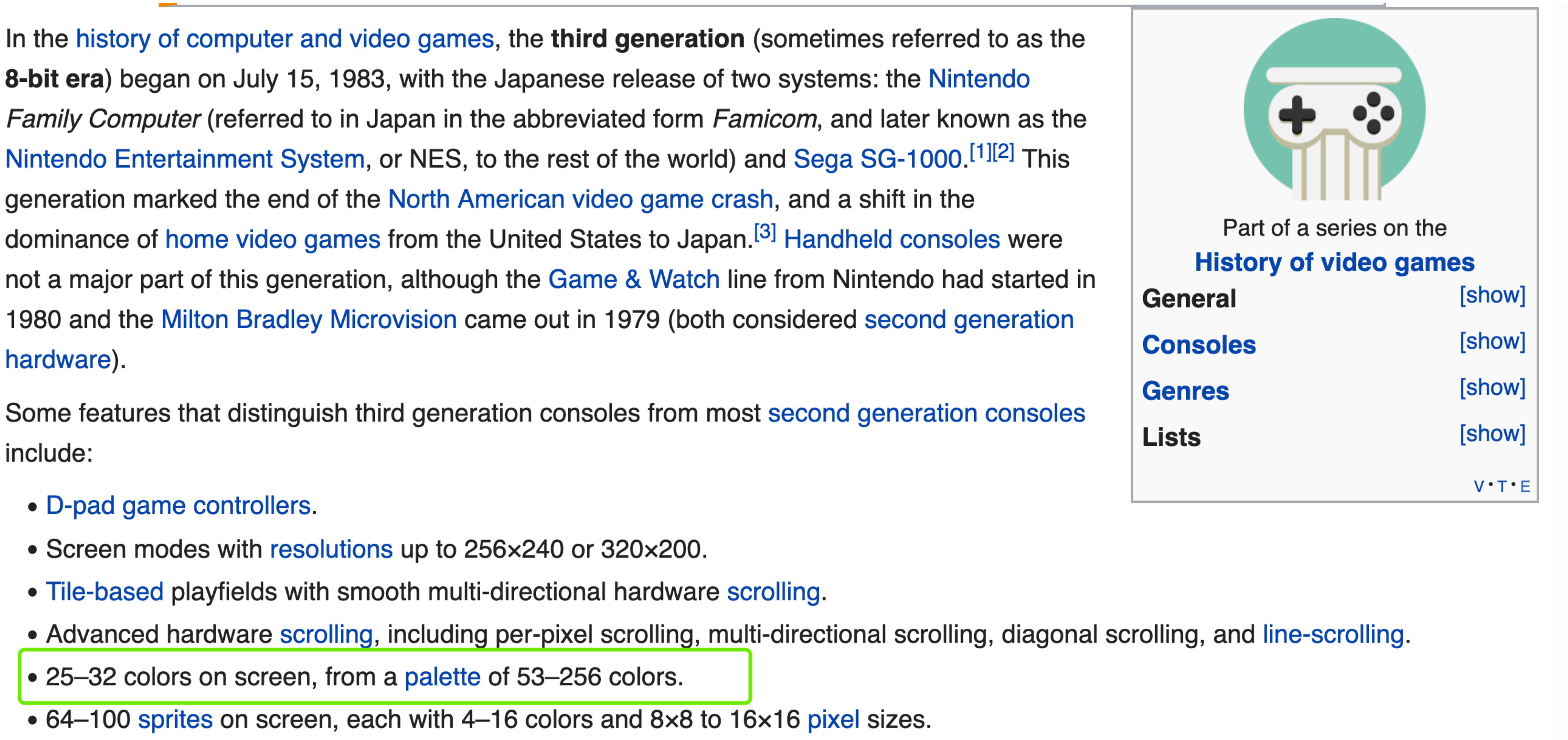
压缩再压缩

80%的功夫都在压缩

怎么压缩

- 从像素点上压缩
- 帧内压缩
- 跨帧压缩

像素点压缩 - Palette



In the history of computer and video games, the **third generation** (sometimes referred to as the **8-bit era**) began on July 15, 1983, with the Japanese release of two systems: the Nintendo *Family Computer* (referred to in Japan in the abbreviated form *Famicom*, and later known as the Nintendo Entertainment System, or NES, to the rest of the world) and Sega SG-1000.[1][2] This generation marked the end of the North American video game crash, and a shift in the dominance of home video games from the United States to Japan.[3] Handheld consoles were not a major part of this generation, although the Game & Watch line from Nintendo had started in 1980 and the Milton Bradley Microvision came out in 1979 (both considered second generation hardware).

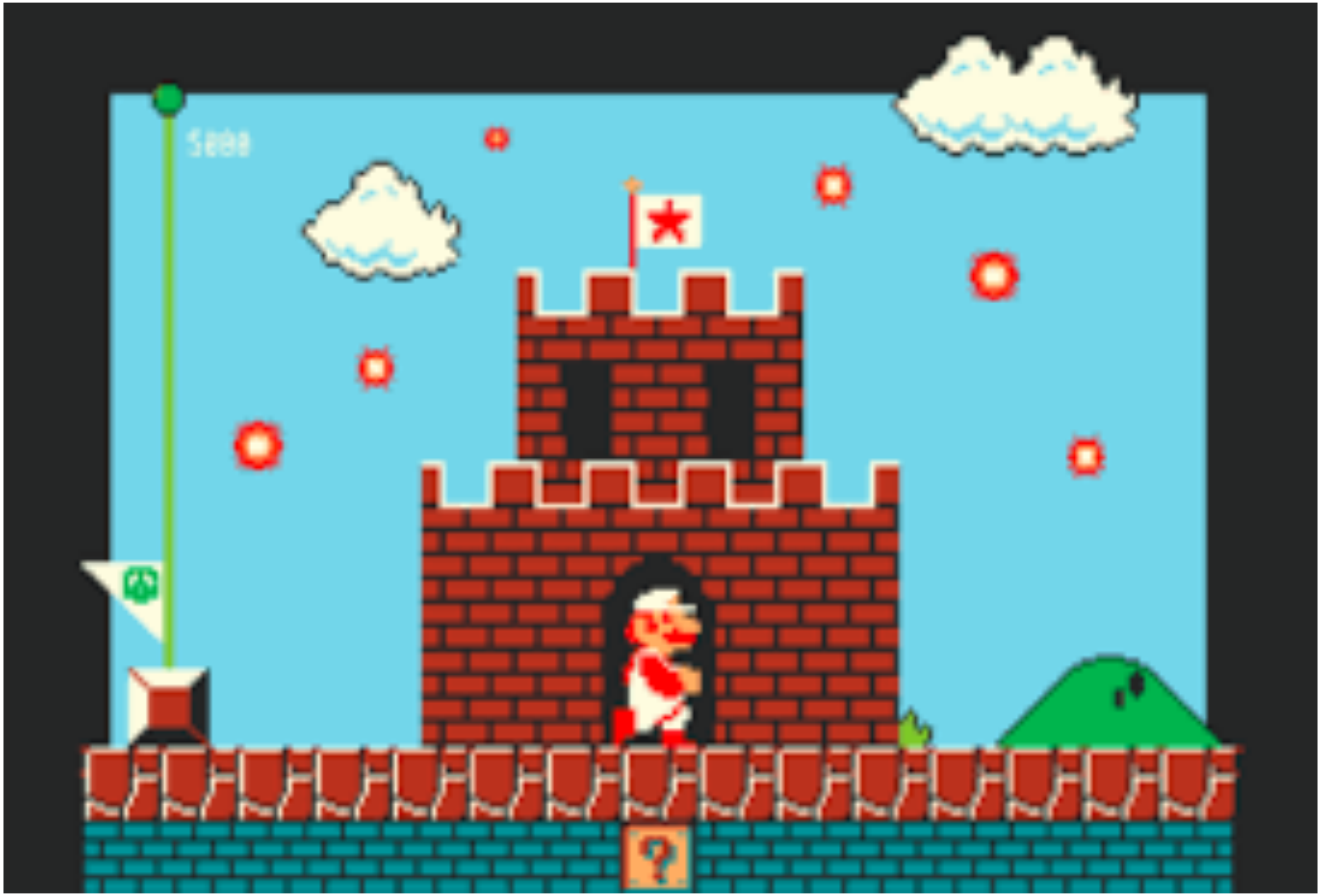
Part of a series on the
History of video games
General [show]
Consoles [show]
Genres [show]
Lists [show]
V • T • E

Some features that distinguish third generation consoles from most second generation consoles include:

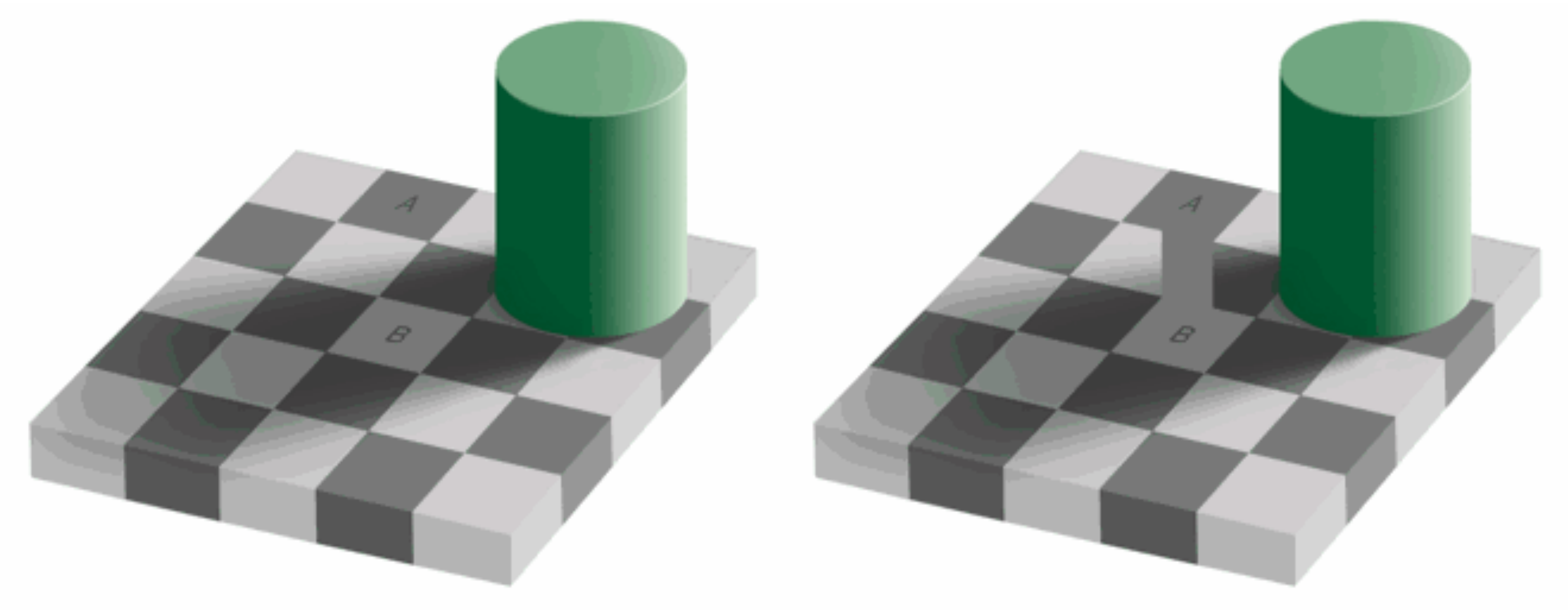
- D-pad game controllers.
- Screen modes with resolutions up to 256×240 or 320×200.
- Tile-based playfields with smooth multi-directional hardware scrolling.
- Advanced hardware scrolling, including per-pixel scrolling, multi-directional scrolling, diagonal scrolling, and line-scrolling.
- 25–32 colors on screen, from a palette of 53–256 colors.
- 64–100 sprites on screen, each with 4–16 colors and 8x8 to 16x16 pixel sizes.

- 不要用3个Byte代表一个点了，干脆约定X个颜色，001代表土黄，002代表土豪金，等等…
- 如果只需要256个颜色，那么1个Byte就够用了，节省了66.6%

但是很丑



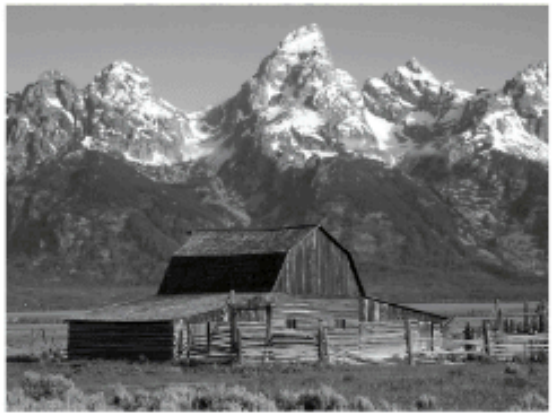
像素点压缩2 - YCbCr



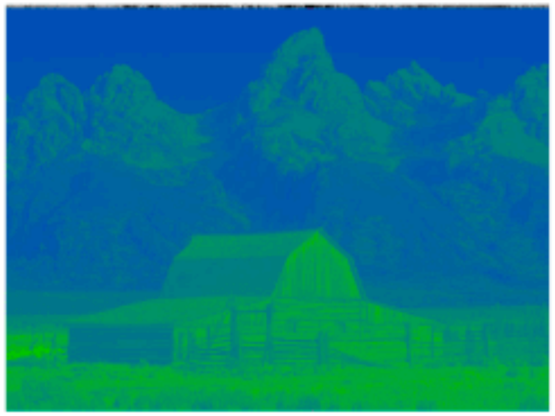
- 人眼对明暗敏感, 对颜色不敏感

像素点压缩2 - YCbCr

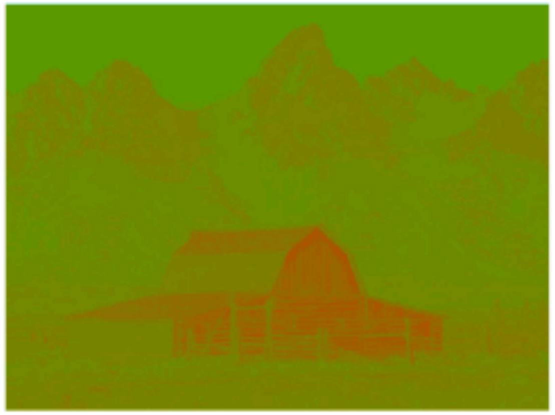
Y (luma)



U (chroma blue)

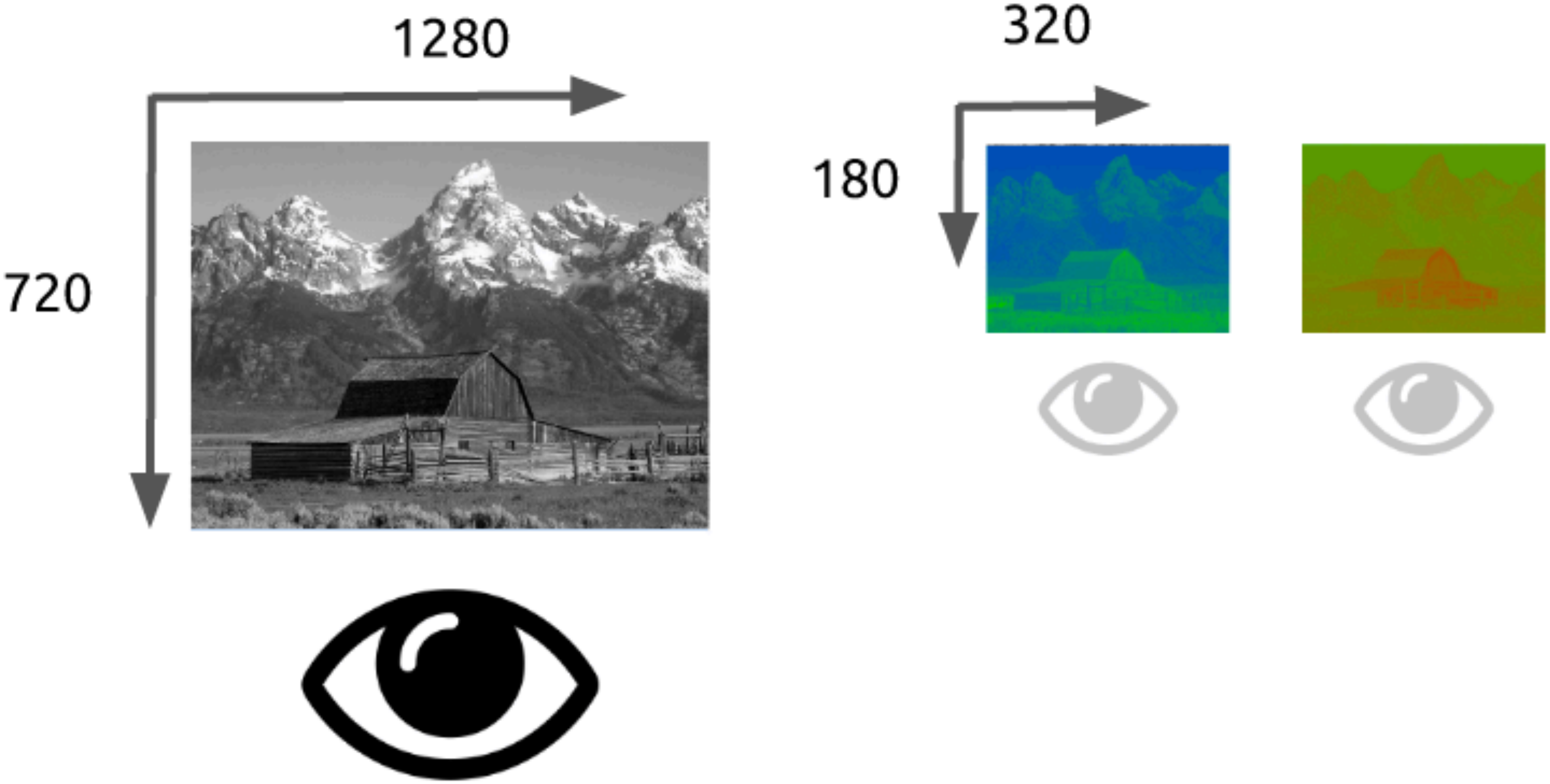


V (chroma red)



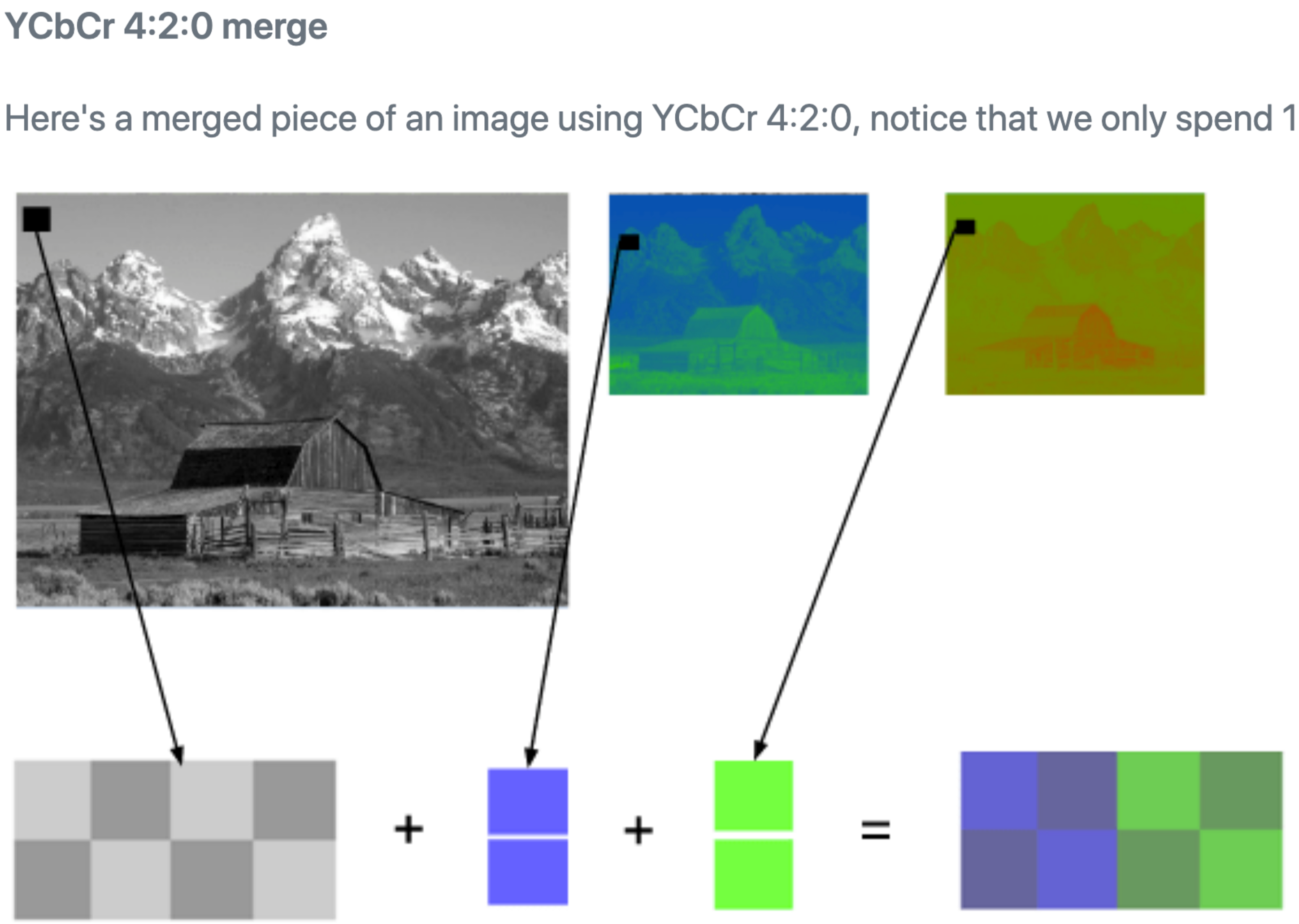
- 不要用RGB, 用YCbCr. 给明暗更多权重, 给颜色少一些权重, 达到省字节+效果差别不大的目的

像素点压缩2 - YCbCr



- Y (明暗) = (R+G+B)/3 (实际会用其他参数值)
- 给明暗更多bits, 给Cr Cb更少bits

像素点压缩2 - YCbCr



YCbCr 4:2:0 merge

Here's a merged piece of an image using YCbCr 4:2:0, notice that we only spend 12 bits per pixel.

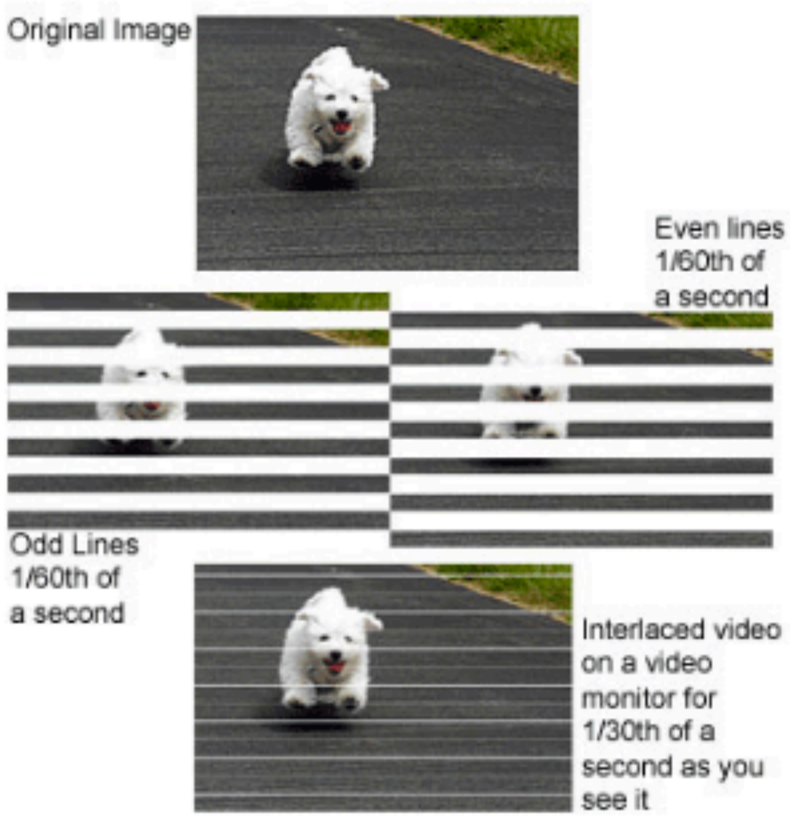
- 分辨率改动放在全图层面

像素点压缩2 - YCbCr

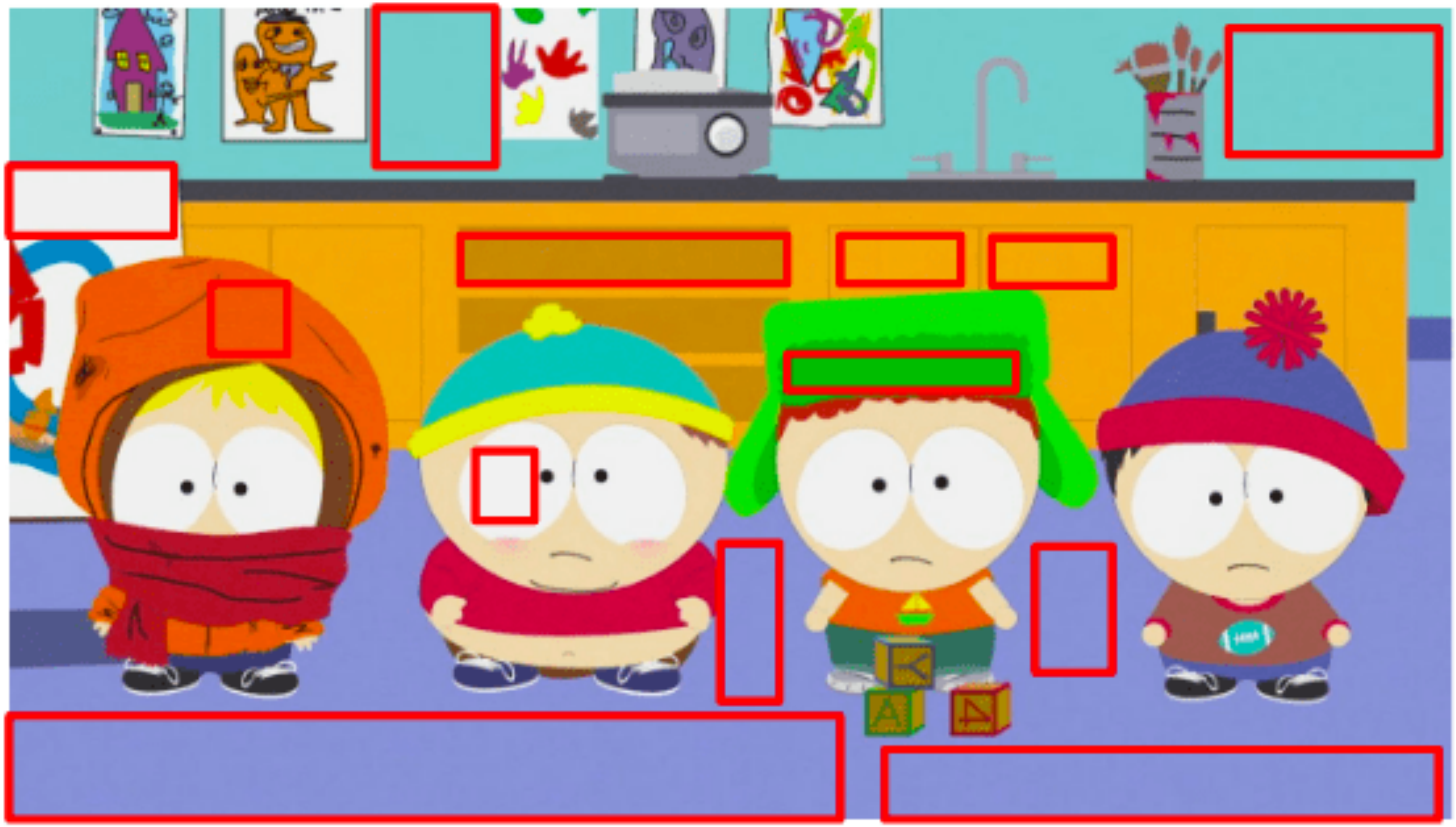
- YCbCr被广泛使用. 为啥是绿屏 - 因为y=255, Cb=0, Cr=0, 结果就是绿色

帧内压缩 - 隔行扫描

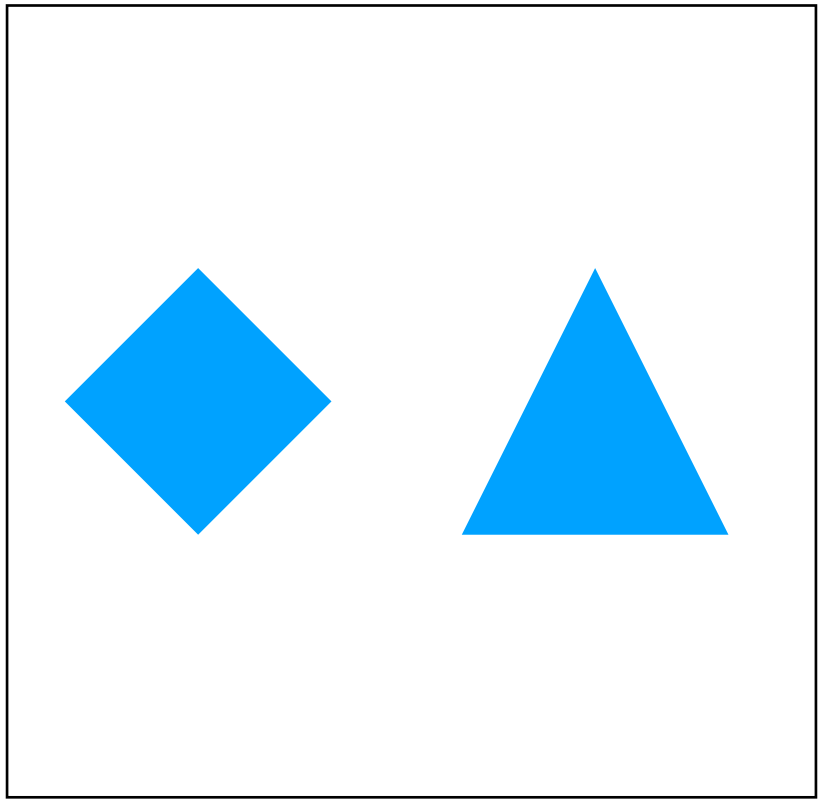
只给你一半, 又省了50%



帧内压缩 - 冗余编码

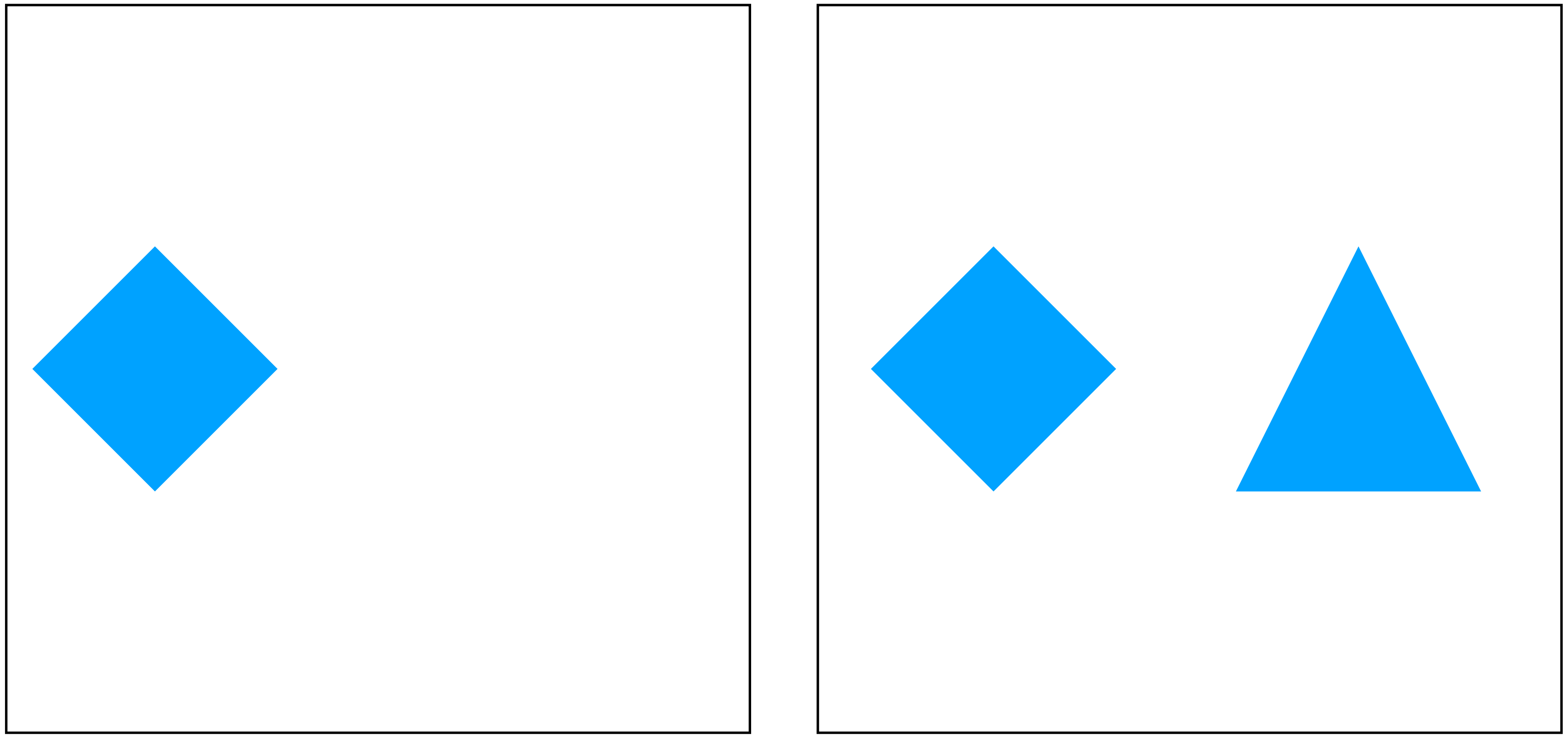


帧间压缩 - 帧类型



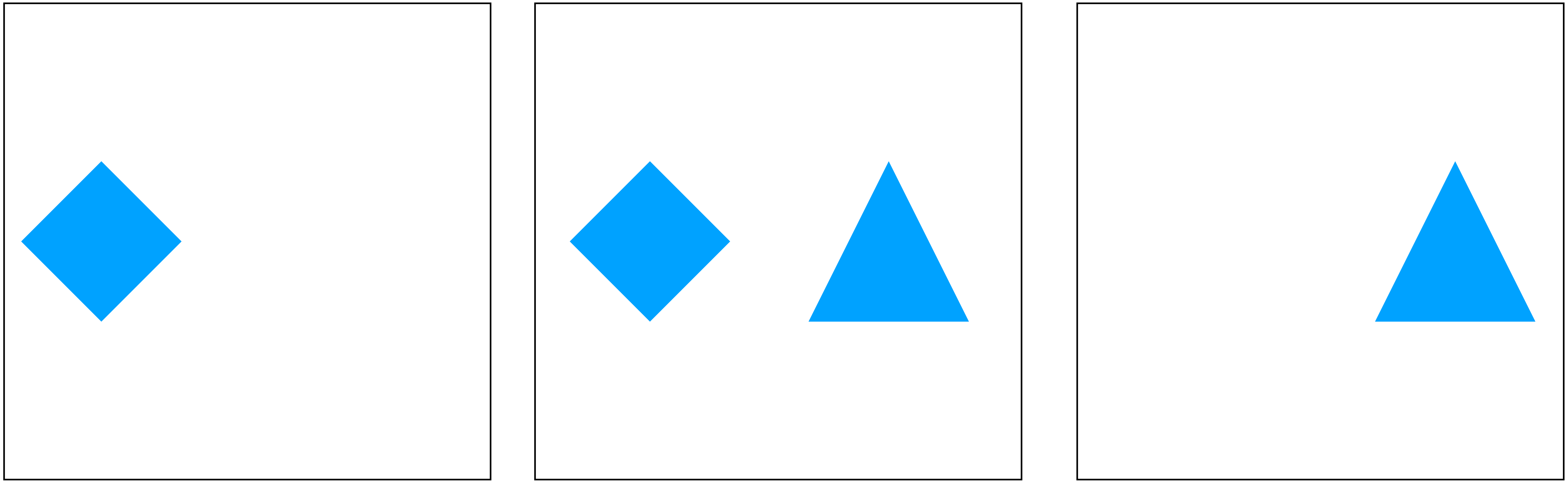
keyframe 关键帧

帧间压缩 - 帧类型



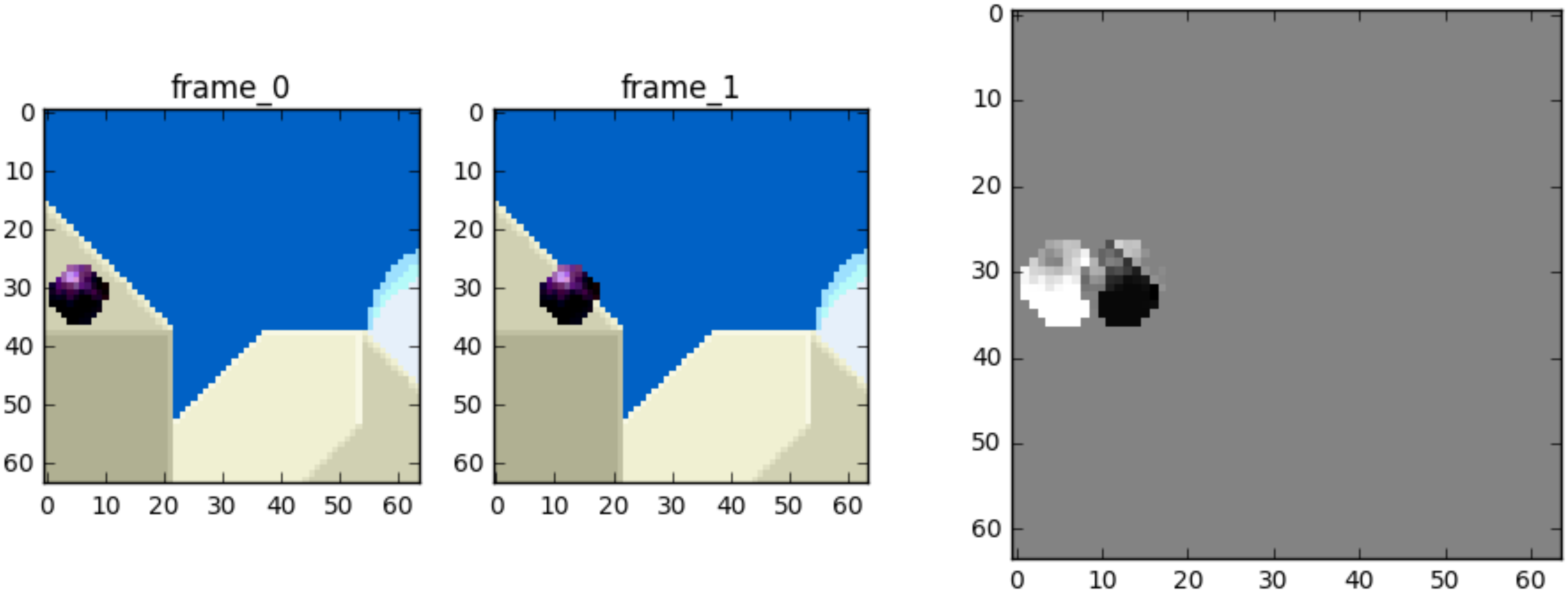
P frame 向前依赖帧

帧间压缩 - 帧类型

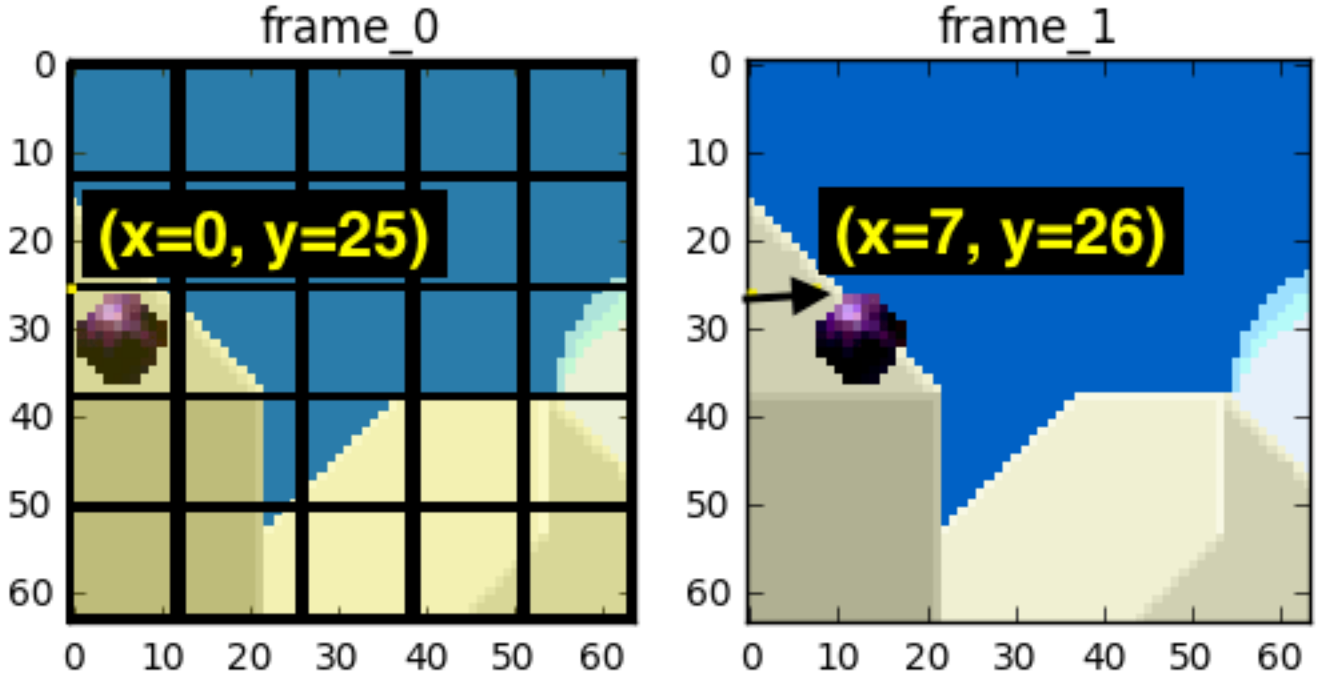


B frame 双向依赖帧

帧间压缩

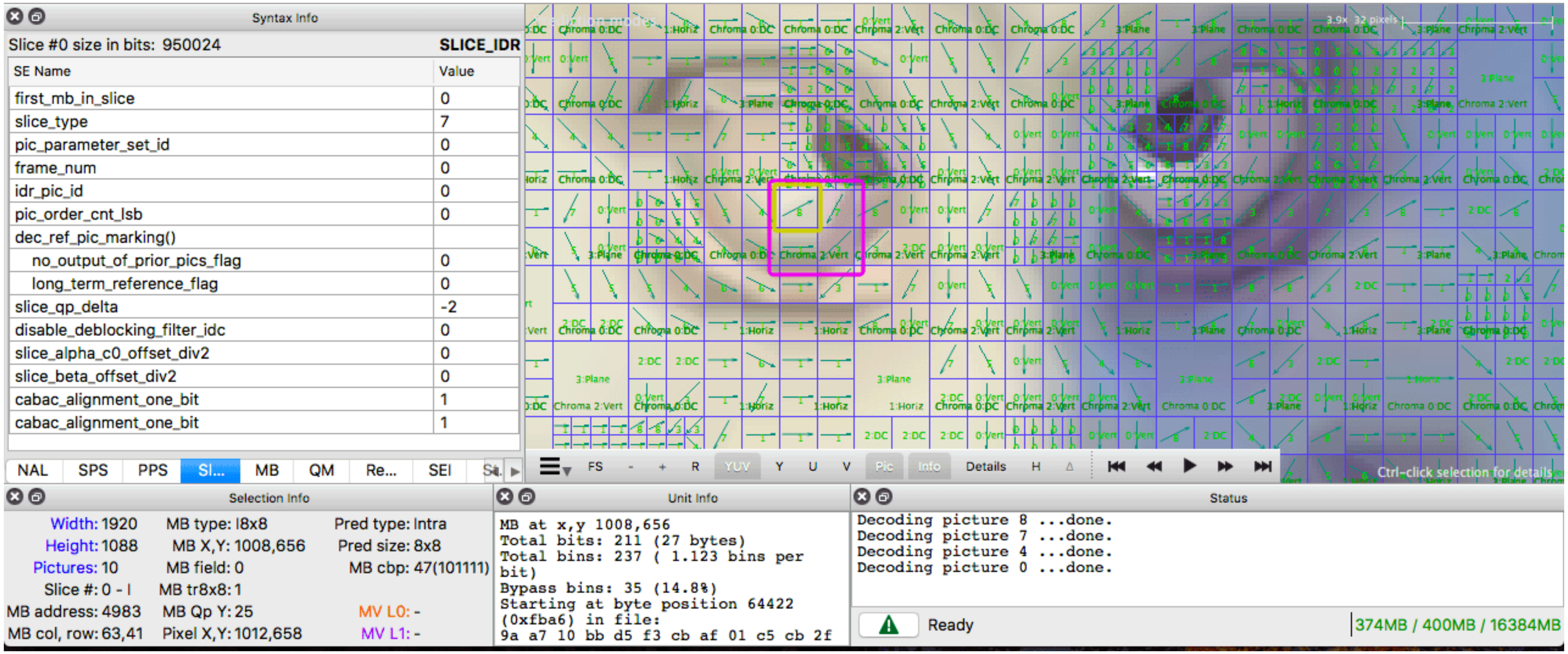


帧间压缩



这里不准确会再算一次diff

帧间压缩



Slice #0 size in bits: 950024 SLICE_IDR

SE Name	Value
first_mb_in_slice	0
slice_type	7
pic_parameter_set_id	0
frame_num	0
idr_pic_id	0
pic_order_cnt_lsb	0
dec_ref_pic_marking()	
no_output_of_prior_pics_flag	0
long_term_reference_flag	0
slice_qp_delta	-2
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	0
slice_beta_offset_div2	0
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

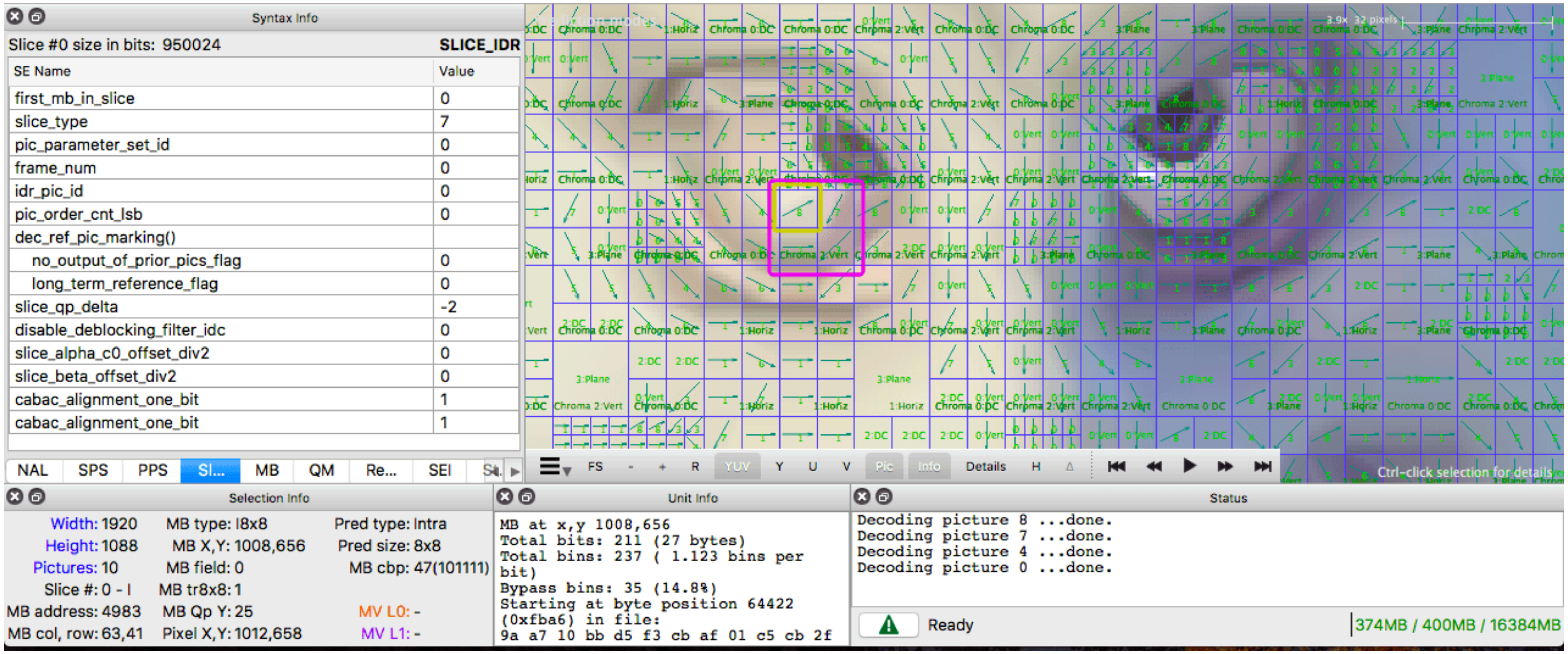
Width: 1920	MB type: I8x8	Pred type: Intra
Height: 1088	MB X,Y: 1008,656	Pred size: 8x8
Pictures: 10	MB field: 0	MB cbp: 47(101111)
Slice #: 0 - I	MB tr8x8: 1	
MB address: 4983	MB Qp Y: 25	MV L0: -
MB col, row: 63,41	Pixel X,Y: 1012,658	MV L1: -

```
MB at x,y 1008,656
Total bits: 211 (27 bytes)
Total bins: 237 ( 1.123 bins per
bit)
Bypass bins: 35 (14.8%)
Starting at byte position 64422
(0xfba6) in file:
9a a7 10 bb d5 f3 cb af 01 c5 cb 2f
```

```
Decoding picture 8 ...done.
Decoding picture 7 ...done.
Decoding picture 4 ...done.
Decoding picture 0 ...done.
```

Ready 374MB / 400MB / 16384MB

帧间压缩



Syntax Info

Slice #0 size in bits: 950024 **SLICE_IDR**

SE Name	Value
first_mb_in_slice	0
slice_type	7
pic_parameter_set_id	0
frame_num	0
idr_pic_id	0
pic_order_cnt_lsb	0
dec_ref_pic_marking()	
no_output_of_prior_pics_flag	0
long_term_reference_flag	0
slice_qp_delta	-2
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	0
slice_beta_offset_div2	0
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

Selection Info

Width: 1920	MB type: I8x8	Pred type: Intra
Height: 1088	MB X,Y: 1008,656	Pred size: 8x8
Pictures: 10	MB field: 0	MB cbp: 47(101111)
Slice #: 0 - I	MB tr8x8: 1	
MB address: 4983	MB Qp Y: 25	MV L0: -
MB col, row: 63,41	Pixel X,Y: 1012,658	MV L1: -

Unit Info

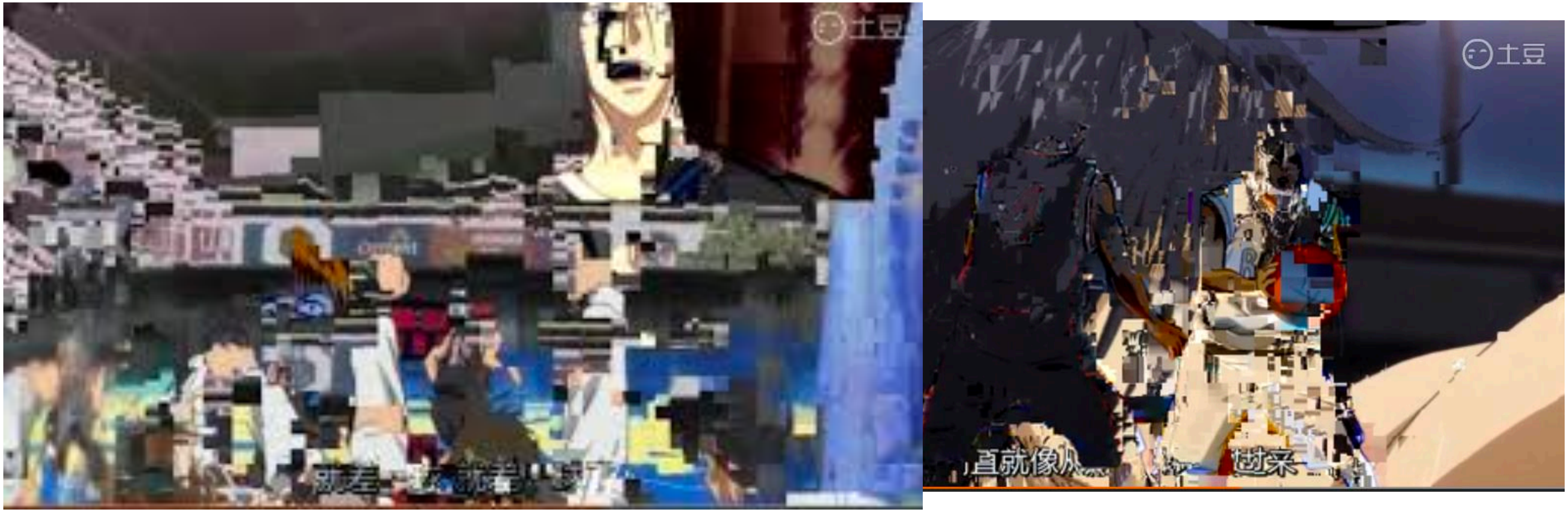
```
MB at x,y 1008,656
Total bits: 211 (27 bytes)
Total bins: 237 ( 1.123 bins per
bit)
Bypass bins: 35 (14.8%)
Starting at byte position 64422
(0xfba6) in file:
9a a7 10 bb d5 f3 cb af 01 c5 cb 2f
```

Status

```
Decoding picture 8 ...done.
Decoding picture 7 ...done.
Decoding picture 4 ...done.
Decoding picture 0 ...done.
```

Ready 374MB / 400MB / 16384MB

帧间压缩



Container格式与Codec编码格式

- Codec是编码解码标准, 之前提到的压缩技巧大部分会在这一层实现

- Container格式是我们更熟悉的, MP4, FLV, MKV等等. 通常规定Box模型, 元数据, 分块, mux/demux等

流媒体直播

- HLS, DASH, MPD, HTTP-FLV等等
- 对称加密/vendor mod
- http://devimages.apple.com/iphone/samples/bipbop/bipbopall.m3u8
-