专题：视频技术的理论与实践

# TPG 图像压缩技术

王诗涛[1]，丁飘[1]，黄晓政[1]，刘海军[1]，罗斌姬[1]，陈新星[1]，吴友宝[1]，王荣刚[2]
（1. 腾讯科技（深圳）有限公司，广东 深圳 518057;
2. 北京大学信息工程学院，广东 深圳 518055）

**摘　要**：TPG（tiny portable graphic）是基于 AVS2 视频编码标准推出的图像压缩技术，该图片格式压缩效率明显高于 JPG、PNG、GIF 等其他传统格式。介绍了 TPG 图像编码技术的编码原理以及技术特点，对比了 TPG 图片格式与传统图片格式的压缩效率，结果显示，TPG 图片格式具有明显编码增益。

**关键词**：TPG；AVS2；图像压缩
**中图分类号**：TN919　　**文献标识码**：A
**doi:** 10.11959/j.issn.1000−0801.2017238

# TPG image compression technology

WANG Shitao[1], DING Piao[1], HUANG Xiaozheng[1], LIU Haijun[1],
LUO Binji[1], CHEN Xinxing[1], WU Youbao[1], WANG Ronggang[2]
1. Tencent Technology (Shenzhen) Co., Ltd., Shenzhen 518057, China
2. School of Electronic and Computer Engineering, Peking University, Shenzhen 518055, China

**Abstract:** TPG(tiny portable graphic) is a new image compression technology based on the video part of AVS2 standard, whose compression efficiency is notably higher than traditional image formats like JPG, PNG and GIF. Theory and feature of TPG image compression technology were introduced. Then, the compression efficiency of TPG and traditional image formats was compared. Results show that TPG has overwhelming advantage.

**Key words:** tiny portable graphic, AVS2, image compression

## 1　引言

随着移动互联网的发展，终端设备的下载流量大幅增长，用户下载流量中，图片流量占据很大比例。图片流量的快速增长，给企业带来非常大的成本压力，很多企业都在尝试各种优化来降低图片流量成本。

以互联网图片为例，对于新闻网页、QQ 空间、朋友圈等应用场景中的图片，通常会在后台将原图转码成更小的图片供用户下载，从而节省流量带宽成本。

如何减小图片大小呢？一种方法是降低图片质量，比如将 JPEG 图片质量由 JPEG80 降低到 JPEG70 甚至更低，这样虽然能节省带宽，但是图片质量也大大下降，很影响用户体验；另一种方法就是采用更高效的图片压缩方法，在保证图片质量的同时将图片压得更小，相比第一种方法，显然这种方法更优。

目前的主流图片格式有 JPEG、PNG 和 GIF

收稿日期：2017−07−15；修回日期：2017−07−26

等。JPEG 诞生于 1992 年，是最常见的图片格式。目前很多学者基于 JPEG 标准对图像压缩算法进行改进，如参考文献[1]，但是由于 JPEG 算法本身的不足，其总体压缩效率并不高。PNG 诞生在 1995 年，它只支持无损压缩，所以它的压缩比是有上限的，尤其是对于颜色和细节丰富的图像，PNG 文件通常会很大。GIF 诞生于 1987 年，随着互联网流行开来。它有很多缺点，比如通常情况下只支持 256 种颜色、透明通道只有 1 bit、文件压缩比不高，它唯一的优势就是支持多帧动画。

为了提升压缩效率，很多公司也推出了自己的图片格式，比如 Firefox 的 APNG、微软的 JPEG XR 以及 Google 的 WebP。这其中以 WebP 的应用最多。2010 年，Google 公司推出了 WebP 图片格式，号称压缩效率相比 JPEG 可以提升 40%，但是根据 Mozilla 的评测结果，WebP 相对 JPEG 的优势并没有宣传中那么明显，反而是 HEVC（又称 H.265）在评测中大幅领先于其他几种格式。另外，根据 JCTVC-I0595 文档中的统计数据，H.265 的 I 帧压缩效率也远远高于 WebP 和 JPEG，比 WebP 高 31%，比 JPEG 高 43%。

AVS2 是国产新一代视频压缩编码标准，其压缩效率与 HEVC 相当，在 I 帧压缩效率方面甚至优于 HEVC。由于 AVS2 高效的压缩效率，基于 AVS2 视频编码标准推出了 TGP 图像压缩技术。

## 2 AVS2 编码标准

AVS2 采用了混合编码框架，整个编码过程包括帧内预测、帧间预测、变换量化、反量化反变换、环路滤波和熵编码等模块[2]。AVS2 编码框架如图 1 所示。

### 2.1 灵活的编码结构划分

AVS2 采用了基于四叉树的块划分结构，包括编码单元（coding unit，CU）、预测单元（prediction unit，PU）和变换单元（transform unit，TU）。一幅图像被分割成固定大小的最大编码单元（largest coding unit，LCU），最大编码单元按照四叉树的方式迭代划分为一系列的 CU。每个 CU 包含一个亮度编码块和两个对应的色度编码块（下文中块单元的大小指亮度编码块）。与传统的宏块相比，基于四叉树的划分结构更加灵活，CU 大小从 8 bit×8 bit 扩展到 64 bit×64 bit[3]。

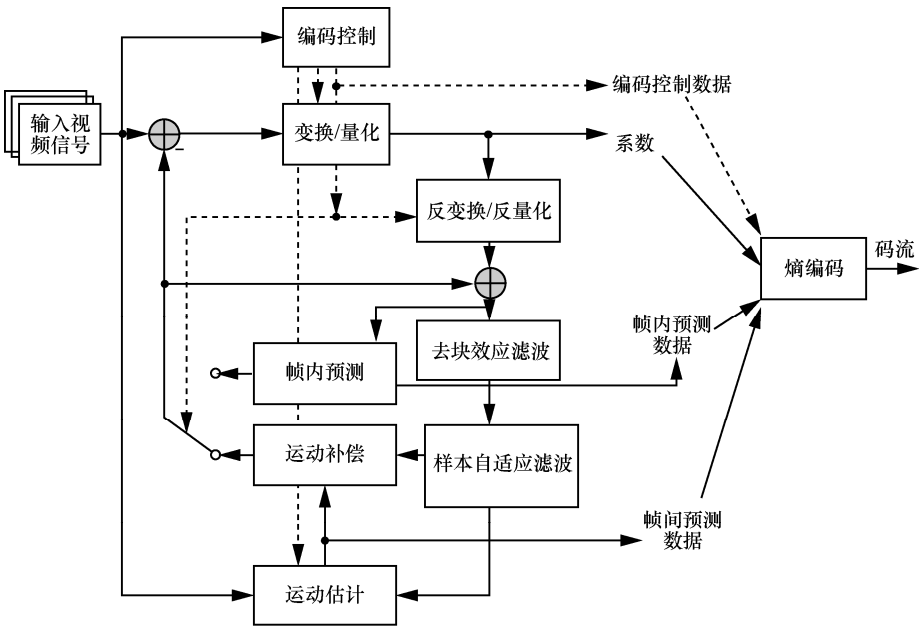



图 1 AVS2 编码框架

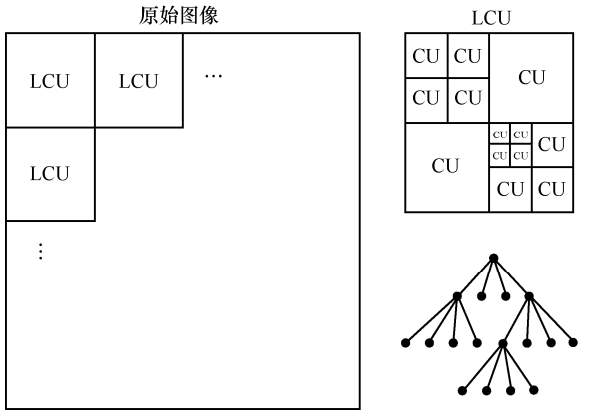



图 2　原始图像、LCU 和 CU 之间的关系以及四叉树的划分结构

PU 规定了 CU 的所有预测模式，是进行预测的基本单元，包括帧内和帧间预测。PU 的最大尺寸不能超过当前所属 CU。在 AVS1 正方形帧内预测块的基础上，增加了非正方形的帧内预测块划分，同时，帧间预测也在对称预测块划分的基础上，增加了 4 种非对称的划分方式。

除了 CU 和 PU，AVS2 还定义了用于预测残差变换和量化的变换单元（TU）。TU 是变换和量化的基本单元，与 PU 一样，定义在 CU 之中。其尺寸的选择与对应的 PU 形状有关，如果当前 CU 被划分为非方形 PU，那么对应的 TU 将使用非方形的划分；否则，使用方形的划分类型。需要注意的是，TU 的尺寸可以大于 PU 的尺寸，但不能超过所在的 CU 尺寸[4]。

### 2.2　帧内预测编码

AVS2 在亮度块的帧内预测编码上设计了 33 种模式，如图 3 所示，包括 DC 预测模式、plane（平面）预测模式、bilinear（双线性）滤波预测模式和 30 种角度预测模式。在色度块上有 5 种模式：DC 模式、水平预测模式、垂直预测模式、双线性插值模式以及新增的亮度导出（derived mode，DM）模式[5]。

### 2.3　帧间预测编码

AVS2 的帧间预测技术在参考帧管理、帧间预测模式和插值方面进行了加强和创新[6,7]。

参考帧管理方面，AVS2 的候选参考帧的最大数量可以到 4 个，以适应多层次的参考帧管理。为了满足多种参考帧管理方式的要求，AVS2 采用了一种多层次的参考帧管理模式。在这个模式中，根据帧与帧之间的参考关系和每个编码图像组（group of picture，GOP）中的帧被分成了多个层次。图 4 展示了 3 种典型的参考关系和分类。

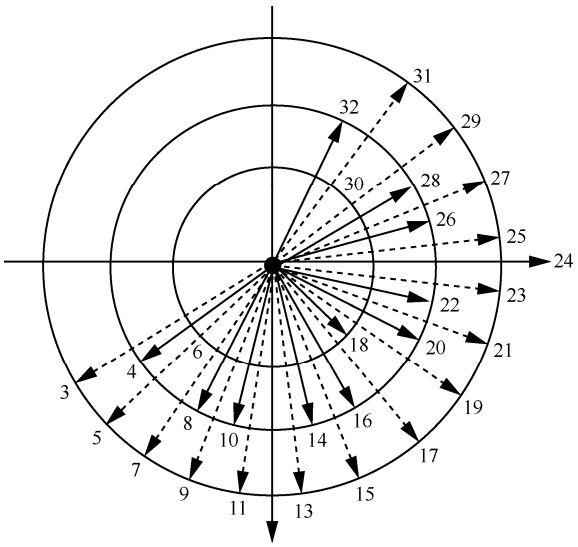



图 3　亮度块帧内预测模式

帧间预测模式方面，AVS2 在 I、P、B 3 种图像类型的基础上，增加了前向多假设预测 F 图像。F 帧图像的编码块可以参考前向两个参考块，相当于 P 帧的双假设预测模式[8]。对于 B 帧，除了传统的前向、后向、双向和 skip（跳过）/direct（直接）模式，新增了对称模式。

插值滤波方面，为了简化运动补偿，AVS2 采用了 8 抽头基于 DCT（discrete cosine transform）的插值滤波器，只需要进行一次滤波，而且支持生成比 1/4 像素更高的运动矢量精度。

### 2.4　变换

AVS2 中的变换编码主要使用整数 DCT。对于 4 bit×4 bit、8 bit×8 bit、16 bit×16 bit、32 bit×32 bit 大小的变换块直接进行整数 DCT。而对于 64 bit×64 bit 大小的变换块，则采用一种逻辑变换，先进行小波变换，再进行整数 DCT。在 DCT 完成后，AVS2 对低频系数的 4 bit×4 bit 块再进行二

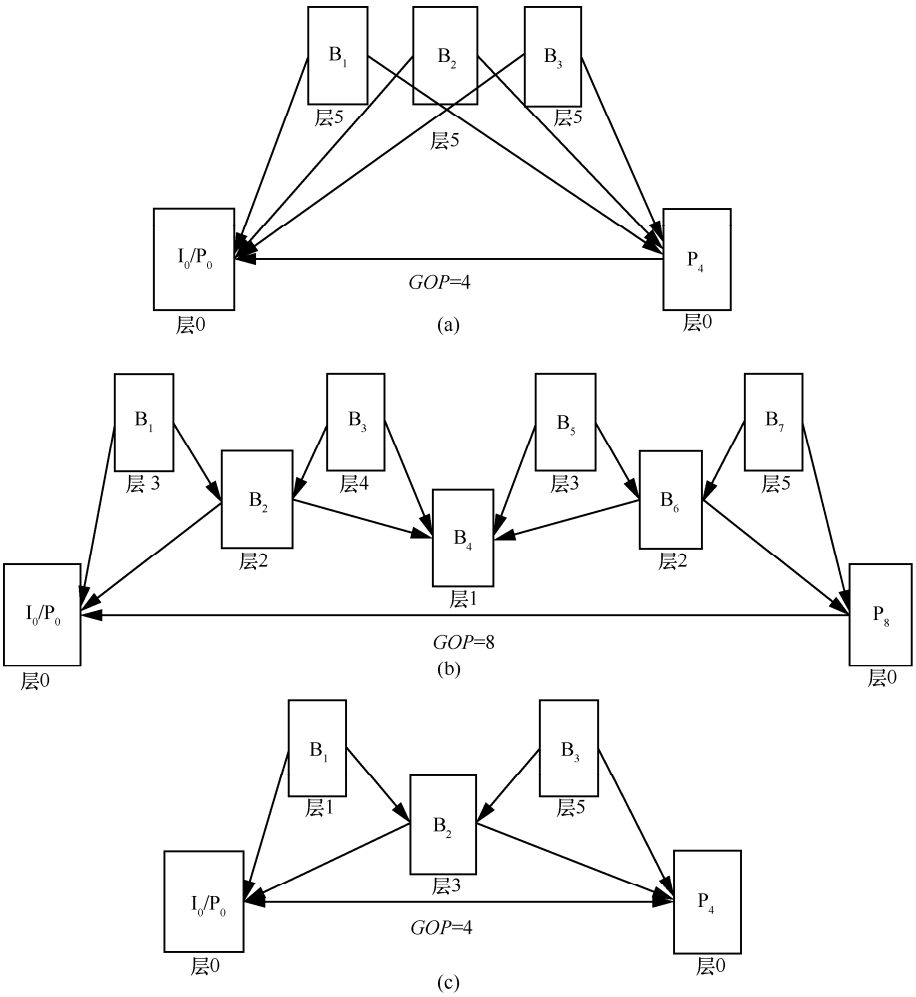



图 4　3 种典型的参考关系和分类

次 4 bit×4 bit 变换，从而进一步降低系数之间的相关性，使能量更集中[9]。

### 2.5　熵编码

AVS2 的熵编码首先将变换系数分为 4 bit×4 bit 大小的系数组（coefficient group，CG），然后根据系数组进行编码和 Zig-Zag 扫描。先编码含有最后一个非零系数的 CG 位置，接着编码每一个 CG，直到 CG 系数都编码完为止，这样可以使得零系数在编码过程中更集中。AVS2 中仍使用基于上下文的二元算术编码和基于上下文的二维变长编码[10]。

### 2.6　环路滤波

AVS2 的环路滤波模块包含 3 个部分：去块滤波、自适应样点偏移和样本补偿滤波。去块滤波的滤波块尺寸为 8 bit×8 bit，首先对垂直边界进行滤波，然后是水平边界。对每条边界根据滤波强度不同选择不同的滤波方式。在去块滤波之后，采用自适应样本偏移补偿进一步减小失真。AVS2 在去块滤波和样本偏移补偿之后又添加了自适应滤波器，一种 7×7 十字+3×3 方形中心对称的维纳滤波，利用原始无失真图像和编码重构图像计算最小二乘滤波器系数，并对解码重构图像进行滤波，降低解码图像中的压缩失真，提升参考图像质量[11,12]。

## 3　TPG 图像压缩技术方案

### 3.1　静态图像编码

一幅静态图像对于视频编码来说，就相当于

一个 I 帧，因此，可以采用 I 帧压缩的方式压缩一张静态图片。不过，传统视频压缩编码针对的是 YUV 空间域的数据，而图像压缩一般是针对 RGB 域数据，所以，编码端进行 I 帧编码之前需要先将原始数据从 RGB 域转换到 YUV 空间域。解码端解码得到 YUV 数据后需要将 YUV 数据转换到 RGB 域。静态图像编码过程如图 5 所示。

对于 RGB 转 YUV，先将 RGB 转换到 YUV444 空间域，然后再对 UV 下采样得到 YUV420 数据作为视频编码器输入，转换过程符合 ITU-R BT.601 标准。

$$\begin{aligned} Y &= ((66\times R+129\times G+25\times B+128)>> 8)+16,\\ U &= ((-38\times R-74\times G+112\times B+128)>>8)+128,\\ V &= ((112\times R-94\times G-18\times B+128)>>8)+128 \end{aligned} \quad (1)$$

解码端先进行 UV 上采样，将 YUV420 数据扩展到 YUV444 空间，然后再将 YUV444 数据转换成 RGB 数据。

$$\begin{aligned} R &= (298\times Y+409\times(Cr-128)+128)>>8,\\ G &= (298\times Y-100\times(Cb-128)+208\times(Cr-128)+128)>>8,\\ B &= (298\times Y+516\times(Cb-128)+128)>>8 \end{aligned} \quad (2)$$

### 3.2　Alpha 通道编码

对于普通 JPG 图片，直接将其 RGB 数据转换成 YUV 数据进行编码即可。但是，对于带 Alpha 通道的 PNG 图片，其输出数据是 RGBA 形式的，采用一个 YUV 空间域数据无法完整描述 RGBA 数据，必须对其做特殊处理。

编码时，将 RGBA 数据分离成 RGB 数据和 Alpha 通道两组数据，然后分别对 RGB 和 Alpha 数据进行编码。对于 RGB 数据，按照前述方法进行编码。对于 Alpha 通道数据，将其视为另一个 Y 通道数据以 YUV400 或者 YUV420（UV 数据置为常量）格式的形式进行编码，然后将 RGB 数据帧和 Alpha 数据帧码流合并生成一个 TPG 图片格式码流。

解码端解码出 RGB 数据帧，输出 YUV 数据，将 YUV 数据转换成 RGB 数据，然后继续解码输出 YUV’数据，其输出的 Y’与前面的 RGB 数据合并输出 RGBA 数据。具体编解码过程如图 6 所示。

### 3.3　动态图片支持

目前，互联网应用中，动态图片越来越多，比如表情图片以及网页中的 GIF 图片等。GIF 图片虽然总量不大，但是由于每个 GIF 图通常比较大，GIF 图在图片总流量中的占比是很大的。如果能减小 GIF 文件大小，不仅能帮企业和用户节省流量带宽，还能减少 GIF 下载的等待时间，提升用户体验。

GIF 编码采用的 256 色编码，GIF 图像帧之间是不会相互参考的，也就是说，GIF 没有利用帧间相关性，其编码效果相比视频压缩效率更低。

TPG 采用视频压缩方式来进行动态图片的压缩。将 GIF、APNG 等解码得到的 RGB 或者 RGBA 数据按照前面所述方法进行编码得到视频帧数据。但是，由于 GIF、APNG 等图像格式的某些特征信息，比如时延信息、透明色等，在视频编码标准里面并没有相关语法元素来描述，因此，需要在视频帧数据前面加入图像帧头，帧头里增

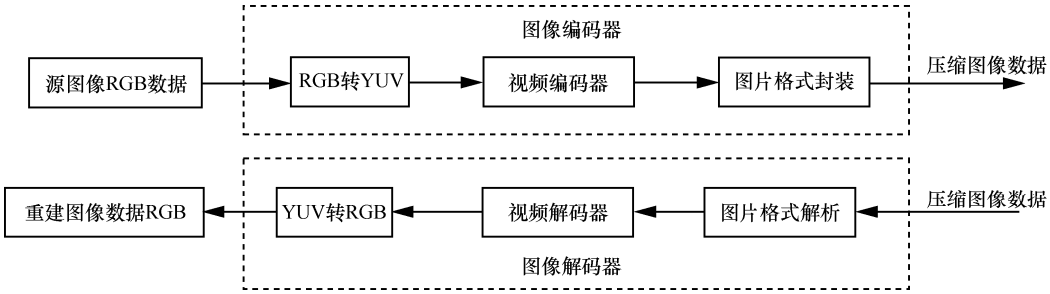



图 5　静态图像编码过程

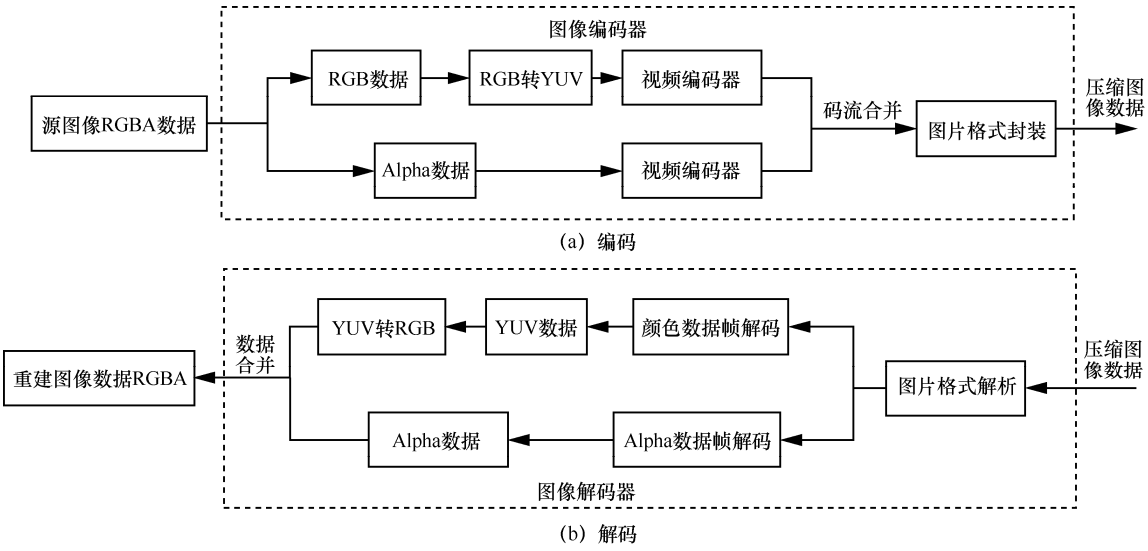



图 6　带 Alpha 通道图像编解码过程

加对这些特征信息的描述。这样就能完整地还原一幅动态图像。具体编码过程如图 7 所示。

### 3.4　TPG 格式介绍

一个完整 TPG 文件的组成结构如图 8 所示，其主要包括图片头信息数据段、图像帧数据段以及透明通道帧数据段，其中，透明通道帧数据段只有在图像有 Alpha 数据时才有。

图 8 中，视频帧数据段是由 AVS2 编码产生的标准码流。在标准视频码流的基础上对其做了图像格式封装，以实现图像的一些特有功能。

图片头信息数据段是 TPG 文件的起始字段，其主要包括图像文件的标识、图像特征信息（比如图像宽高及透明度等）、图像辅助信息（比如 EXIF 信息等）。

图像帧数据段是图像 RGB 数据编码产生的码流数据，包括图像帧头数据段和视频帧数据段两部分，图像帧头数据段用来表示图像序列中每一幅图像的特征信息，每个图像帧都带有一个图像帧头。

透明通道帧数据段是图像 Alpha 通道数据编码产生的码流数据，它包括透明通道帧头数据段和视频帧数据段两部分。透明通道帧头数据段用来描述 Alpha 通道数据帧的特征信息，每个透明通道帧都带有一个透明通道帧头。

### 3.5　TPG 解码

TPG 解码过程主要包括文件头解析、AVS2 解码、YUV 色度分量上采样以及 YUV 到 RGB 颜

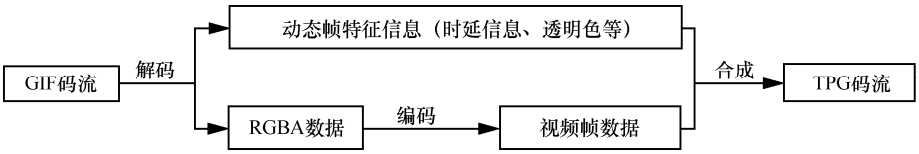



图 7　动态图片格式编码过程

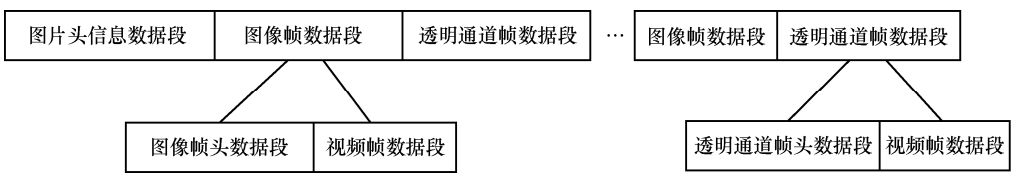



图 8　TPG 文件数据结构

色空间转换几个步骤，其流程如图 9 所示。

3.5.1　色度分量上采样

当 AVS2 解码输出是 YUV4:2:0 格式时，需要对 UV 分量进行上采样得到 YUV4:4:4 格式，然后再进行 YUV 到 RGB 的转换。转换过程如图 10 所示。*src*(*x*,*y*)代表输入的 U 或者 V 分量，*dst*(*x*,*y*)代表上采样后得到的 U 或者 V 分量。假定原始 U、V 分量的宽为 *W*，高为 *H*，则经过下采样得到的新的 U’、V’分量的宽为 2*W*，高为 2*H*。

边界像素和非边界像素采用不同的上采样算法，下面分别介绍。

（1）*dst*(*x*,*y*)非边界像素

非边界像素每个上采样的值由输入图像对应位置相邻 4 个像素共同决定。具体计算规则如下：

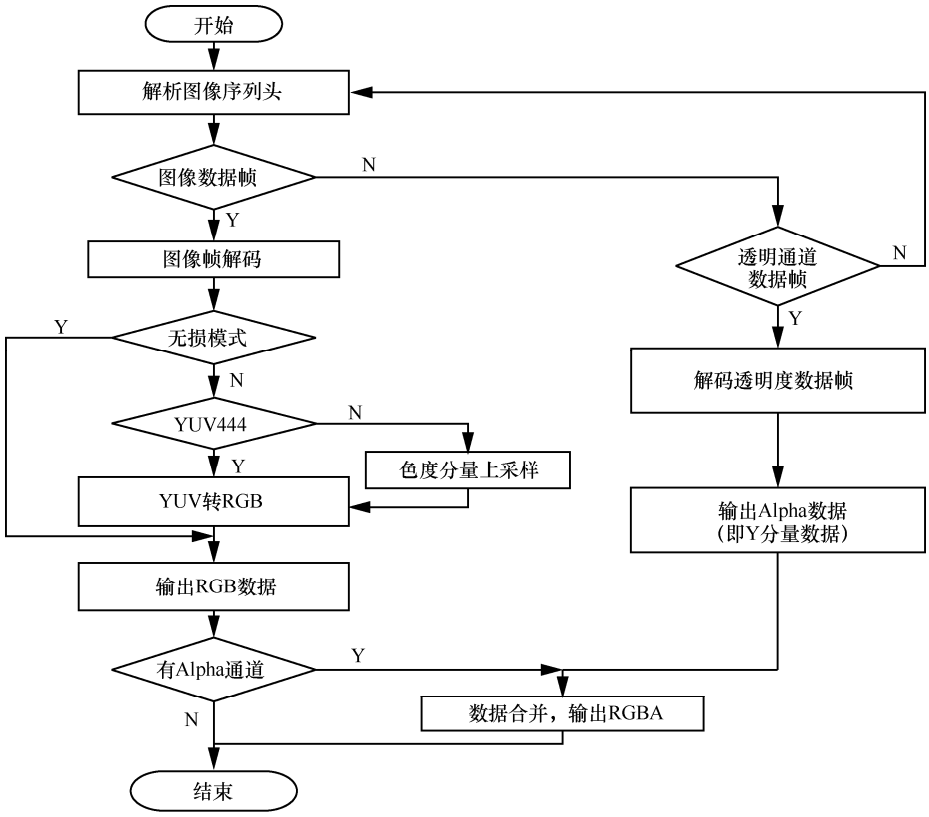



图 9　TPG 解码流程

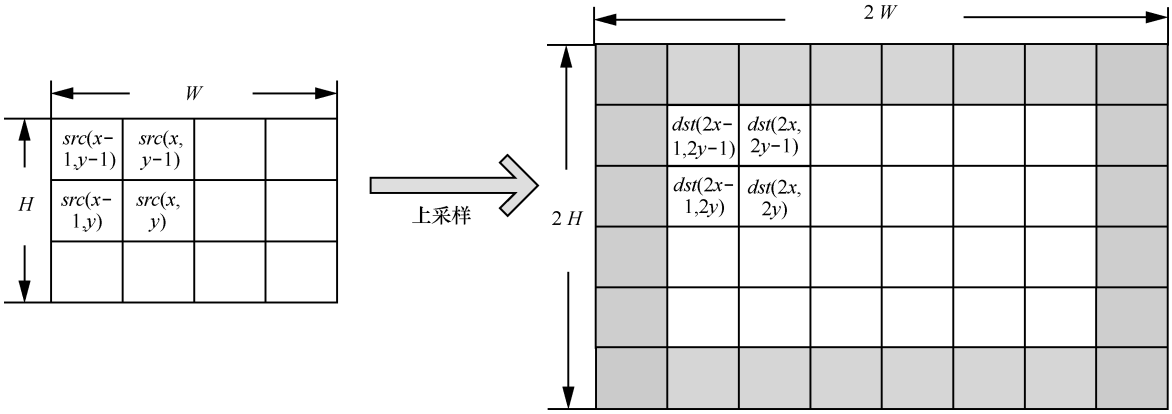



图 10　色度分量上采样示意

$dst(2x, 2y) = (src(x-1, y-1) + 3src(x, y-1) + 3src(x-1, y) + 9src(x, y) + 8) >> 4, 0 < x < W, 0 < y < H$ （3）

$dst(2x-1, 2y) = (3src(x-1, y-1) + src(x, y-1) + 9src(x-1, y) + 3src(x, y) + 8) >> 4, 0 < x < W, 0 < y < H$ （4）

$dst(2x, 2y-1) = (3src(x-1, y-1)+ 9src(x, y-1)+ src(x-1, y) + 3src(x, y)+8) >> 4, 0 < x < W, 0 < y < H$ （5）

$dst(2x-1, 2y-1) = (9src(x-1, y-1)+3src(x, y-1)+3src(x-1, y)+src(x, y)+8) >> 4, 0<x<W, 0<y<H$ （6）

（2）$dst(x,y)$为边界像素

对于 4 个角点位置的像素值由输入图像的 4 个角点像素决定：

$$dst(0, 0) = src(0, 0) \quad (7)$$

$$dst(0, 2H-1) = src(0, H-1) \quad (8)$$

$$dst(2W-1, 0) = src(W-1, 0) \quad (9)$$

$$dst(2W-1, 2H-1)=src(W-1, H-1) \quad (10)$$

第一行和最后一行除角点外的像素，其值由输入图像的第一行和最后一行对应位置的两个相邻像素点决定。

$dst(2x, 0) = (src(x-1, 0) + 3src(x, 0) + 2) >> 2，0 < x < W$ （11）

$dst(2x-1, 0) = (3src(x-1, 0) + src(x, 0) + 2) >> 2，0 < x < W$ （12）

$dst(2x, 2H-1) = (src(x-1, 2H-1) + 3src(x, 2H-1) + 2) >> 2, 0<x<W$ （13）

$dst(2x-1,2H-1)=(3src(x-1, 2H-1)+src(x, 2H-1) + 2) >>2，0<x<W$ （14）

第一列和最后一列除角点外的像素，其值由输入图像的第一列和最后一列对应位置的两个相邻像素点决定。

$dst(0, 2y-1) = (src(0, y-1)+ 3src(0, y) + 2) >> 2，0 < y < H$ （15）

$dst(0, 2y) = (3src(0, y) + src(0, y-1) + 2) >> 2，0 < y < H$ （16）

$dst(2W-1,2y-1)= (src(2W-1, y-1) + 3src(2W-1, y) + 2) >> 2，0 < y < H$ （17）

$dst(2W-1, 2y) = (3src(2W-1, y) + src(2W-1, y-1) + 2) >> 2，0 < y < H$ （18）

### 3.5.2 YUV 与 RGB 颜色空间转换

由于图像原始数据都是 RGB 域数据，所以需要将解码输出数据由 YUV 颜色空间转换到 RGB 颜色空间。颜色空间转换根据 YUV 标准制式的不同，其转换过程也不一样。采用 ITU-R BT.601 标准的 YUV 值域范围，其 YUV 转 RGB 过程如下：

$R = Clip3( 0, 255, ( 298Y + 409(Cr - 128) + 128 ) >> 8)$，

$G = Clip3( 0, 255, ( 298Y - 100(Cb - 128) + 208(Cr - 128) + 128 ) >> 8)$，

$B = Clip3( 0, 255, ( 298Y + 516(Cb - 128) + 128 ) >> 8)$ （19）

## 4 TPG 评测数据

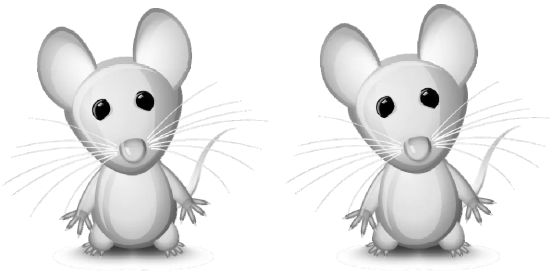
将 TPG 图片格式与传统图片格式进行了对比，接下来将介绍相关的对比统计数据。

### 4.1 主观质量对比

分别将 JPEG、PNG 和 GIF 图片格式转换为 TPG 格式，然后再将 TPG 解码还原成 PNG（如果是动态图片格式则还原出 GIF）图片对比其主观质量。对比效果如图 11~图 13 所示。

（a）源 JPG 图片（43 KB） （b）TPG 图片（24 KB）

图 11 静态图片格式效果对比



（a）源 PNG 图片（292 KB） （b）TPG 图片（46 KB）

图 12 带 Alpha 通道静态图片格式效果对比


（a）源 GIF 图片（1.79 MB）


（b）TPG 动态图片（153 KB）

图 13　动态图片格式效果对比

图 11~图 13 分别对比了 JPEG、PNG 以及 GIF 3 种图片格式和 TPG 的主观效果。可以看出，TPG 图片与源图肉眼看几乎没有区别。但是相比 JPEG 源图，TPG 减小了 44%以上；相比 PNG 源图，大小减小了 80%以上；相比 GIF 源图，大小减小了 90%以上。

### 4.2　客观质量对比

除了主观质量方面的对比，还在客观质量方面对这几种图片格式进行了对比。对比了在 PSNR 对齐的情况下，TPG 与 JPEG 的压缩效率，具体结果如图 14~图 16 所示。

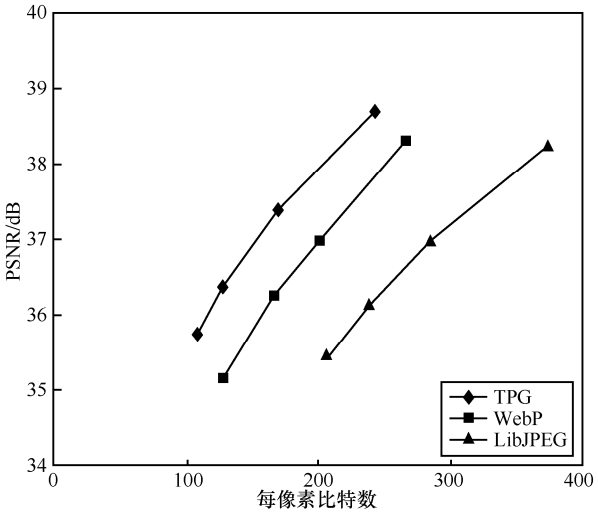



图 14　TPG & WebP & LibJPEG 编码效率对比

如图 14 所示，TPG 的编码效率比 WebP 高 23.5%，比 LibJPEG 高 46.8%。如图 15 所示，Windows 平台下 TPG 的编码速度是 WebP 的 1.6 倍，是 LibJPEG 的 37.6 倍；Linux 平台下 TPG 的编码速度是 WebP 的 2.4 倍，是 LibJPEG 的 38.4 倍。如图 16 所示，Linux 平台下 TPG 的解码速度是 WebP 的 0.7 倍，是 LibJPEG 的 6 倍；iOS 平台下 TPG 的解码速度是 WebP 的 2.2 倍，是 LibJPEG 的 2.6 倍。

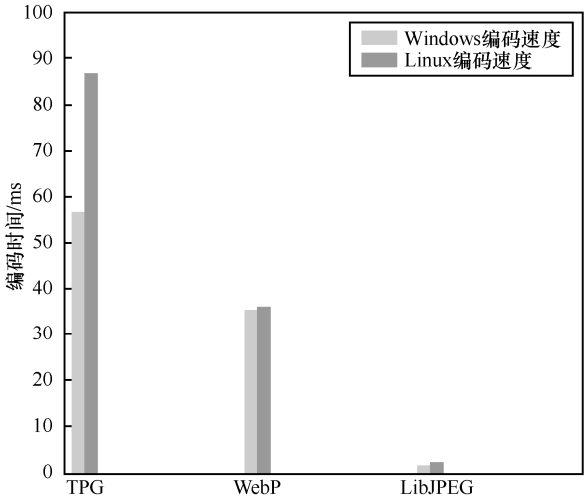



图 15　TPG & WebP & LibJPEG 编码速度对比

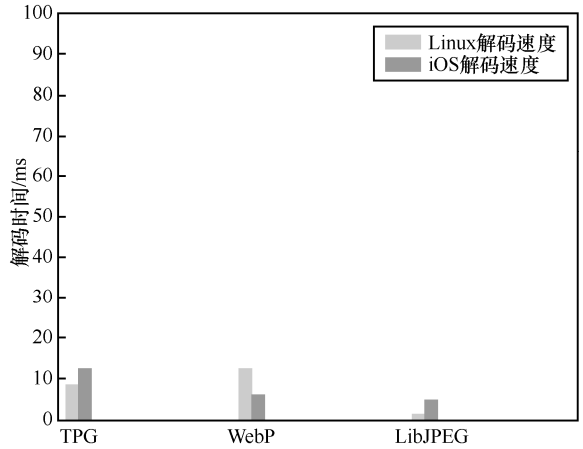



图 16　TPG & WebP & LibJPEG 解码速度对比

## 5　结束语

本文提出了一种新的图片格式 TPG，TPG 基于 AVS2 视频编码标准，通过视频编码的方法来实现图像压缩。TPG 的压缩效率明显高于 JPEG、GIF、PNG 等传统格式，采用 TPG 能帮企业和用户节省大量图片流量带宽。同时，减少了用户下载图片的等待时间，具有重要意义。现今，越来越多的企业开始关注如何节省图片流量，TPG 为省流量提供了一套很好的解决方案。TPG 推出后，

也得到了业界的广泛关注，相信未来 TPG 一定会得到很好的推广和应用。

## 参考文献：


[1] 徐妮妮, 李晨光. 基于 JPEG 标准的 16 bit 图像有损压缩应用[J]. 电信科学, 2016, 32(4): 103-108.
XU N N, LI C G. Application of 16 bit image lossy compression based on the JPEG standard[J]. Telecommunications Science, 2016, 32(4): 103-108.

[2] 黄铁军. AVS2 标准及未来展望[J]. 电视技术, 2014, 38(22): 7-10.
HUANG T J. AVS2 standards and future prospects[J]. Video Engineering, 2014, 38(22): 7-10.

[3] 周芸, 郭晓强, 王强. AVS2 视频编码关键技术[J]. 广播电视信息, 2015(9): 18-21.
ZHOU Y, GUO X Q, WANG Q. AVS2 video coding key technology[J]. Radio & Television Information, 2015(9): 18-21.

[4] 林琪. AVS2 帧间预测技术及快速算法研究[D]. 上海: 上海大学, 2015.
LIN Q. Research on AVS2 inter - frame prediction technology and fast algorithm[D]. Shanghai: Shanghai University, 2015.

[5] PIAO Y, CHEN J, LEE S, et al. Intra coding of AVS2 video coding standard[C]//IEEE International Conference on Multimedia and Expo Workshops, July 14-18, 2014, Chengdu, China. New Jersey: IEEE Press, 2014: 1-5.

[6] HE Z, YU L, ZHENG X, et al. Framework of AVS2-video coding[C]//IEEE International Conference on Image Processing, Sept. 15-18, 2013, Melbourne, VIC, Australia. New Jersey: IEEE Press, 2014: 1515-1519.

[7] MA S, HUANG T, READER C, et al. AVS2? making video coding smarter[J]. IEEE Signal Processing Magazine, 2015, 32(2): 172-183.

[8] SHAO Z, YU L. Multi-directional skip and direct modes design in bi-predictive slices for AVS2 standard[C]//2014 IEEE International Conference on Multimedia and Expo Workshops (ICMEW), July 14-18, 2014, Chengdu, China. New Jersey: IEEE Press, 2014: 1-5.

[9] 韩宏凯. AVS2 帧内预测编码的 CU 划分快速算法及 RDO 优化研究[D]. 成都: 西南交通大学, 2016.
HAN H K. Research on fast algorithm and RDO optimization of CU partition in AVS2 intra prediction code[D]. Chengdu: Southwest Jiaotong University, 2016.

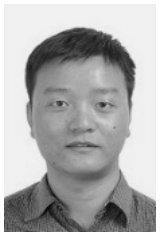
[10] GAO W, MA S. Advanced video coding systems[M]. Berlin: Springer Publishing Company, 2015.

[11] FAN K, WANG R G, WANG Z Y, et al. iAVS2: a fast intra encoding platform for IEEE 1857.4[J]. IEEE Transactions on Circuits and Systems for Video Technology, Early Access, 2016(99): 1.

[12] WANG Z Y, WANG R G, FAN K, et al. uAVS2—fast encoder for the 2nd generation IEEE 1857 video coding standard[J]. Signal Processing: Image Communication, 2017, 53(5): 13-23.

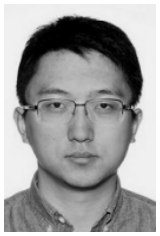


[作者简介]



**王诗涛**（1983−），男，腾讯科技（深圳）有限公司高级工程师，主要研究方向为视频编解码、图像压缩、实时视频通信。

**丁飘**（1983−），男，腾讯科技（深圳）有限公司高级工程师，主要研究方向为实时视频通信、网络技术。



**黄晓政**（1986−），男，腾讯科技（深圳）有限公司高级工程师，主要研究方向为视频编解码和视频通信。

**刘海军**（1988−），男，腾讯科技（深圳）有限公司高级工程师，主要研究方向为实时视频通信以及视频编解码算法。

**罗斌姬**（1989−），女，腾讯科技（深圳）有限公司工程师，主要研究方向为视频编解码、图像编解码、实时通信。

**陈新星**（1982−），男，腾讯科技（深圳）有限公司高级工程师，主要研究方向为实时视频通信以及视频编解码算法。

**吴友宝**（1990−），男，腾讯科技（深圳）有限公司助理工程师，主要研究方向为音视频质量评测工具开发。

**王荣刚**（1976−），男，博士，北京大学信息工程学院副教授，主要研究方向为视频编码、虚拟现实以及图像增强。