

一种新的基于基准线的形状编码

龚声蓉 陆悌亮 沈项军 刘纯平

(苏州大学计算机科学与技术学院, 苏州 215006)

摘要 在图像编码系统中,为了获得较好的解码质量,提出了一种新的基于基准线的形状编码算法。该算法首先采用1维数据描述2维形状信息,并借助形状轮廓来抽取距离集和拐点;然后采用DPCM对距离集和拐点进行编码,从而有效地节约了码流。实验结果表明,该算法不仅能够适应各种特殊的形状边界走向,具有很好的重构效果,而且能有效克服基于块的形状编码所产生的阶梯效应问题。

关键词 距离集 拐点 基准线 形状编码

中图法分类号:TN919.81 文献标识码:A 文章编号:1006-8961(2008)04-0605-07

A New Baseline-based Shape Coding

GONG Sheng-rong, LU Ti-liang, SHEN Xiang-jun, LIU Chun-pin

(Computer Science and Technology School, Soochow University, Suzhou 215006)

Abstract A new baseline-based shape coding algorithm is designed in this paper. Distance set and turning point are chosen from the shape contour, and coded by DPCM. So the 2-dimensional data can be described by 1-dimensional data, and K K reduce the bit streams effectively. The algorithm can adapt to every particular boundary trend, and gives a better reconstructed effect. The reconstructed shape only has a geometrical error, and can overcome the ladder effect which always appears in the block-based shape coding.

Keywords distance set, turning point, baseline, shape coding

1 引言

MPEG-4是由运动图像专家小组(MPEG)负责制定的视、音频编码以及传输的多媒体国际标准。MPEG-4在研究低码率编码的同时,将重点放在人们感兴趣的图像目标的交互性和可操作性上。它是第1个让用户可在接收端对画面进行操作和进行交互式访问的标准,并且它还具有低码率、高灵活性、兼容性、可扩展性和强抗错性等优点。VOP(video object plane)的形状信息是MPEG-4区别于以前视频编码标准的最显著特点,也是MPEG-4实现交互性和可操作性的关键技术。同时,它也渗透于运动预测、运动补偿和纹理编码等部分,用于改进运动和纹理编码的效率。

VOP形状编码存在以下两种典型的方法:(1)基于位图的形状编码,这种方法来源于传统的纹理编码方法,它首先将形状矩阵分块,然后用运动估计消除时间冗余,同时用不同的帧内编码方法消除空间冗余,最后采用不同的帧内编码方法形成不同形状编码方法;(2)基于边缘轮廓的形状编码,该方法有效地将形状的压缩与边缘的跟踪联系起来,从而更有利于提高编码性能。

MPEG-4标准发布以来,形状编码的研究取得了较大进展,尤其是在形状编码的算法改进以及新算法的提出方面取得了一定的成果。如Paulo Nunes等人提出了改进的链编码——基于网格的链编码^[1],该方法不仅提高了链编码的效率,而且不会对对象区域造成误差,也不会出现将一个形状识别成多个形状等错误。Chung等人提出了改

进的基于顶点的形状编码^[2],其比传统的基于顶点编码方法节约了17%~29%的码流,但算法的时间复杂度较大,不适于实时应用。Luo提出了基于图像的形状编码^[3],该方法结合图像进行形状编码,编码时间虽花费比较大,而解码时间花费则较小,因此,该方法比较适用于网络应用,且可以一次编码,多次下载解码。为了减少VOP帧内形状编码的比特量, Lee等人提出了基于基准线的形状编码方法^[4],由于该方法采用1维数据来表达VOP的帧内形状,解码结果仅仅具有几何误差,因而能有效克服基于块的形状编码产生的阶梯效应问题。该方法虽然思路新颖、独特,然而其核心算法——距离集和拐点的采样算法只有特别光滑且较规则的形状才可以正确编码,而对于本文第2部分中提到的几种情况的考虑尚显不足^[5,6]。事实上,这种理想的视频对象形状在现实的视频处理中存在的机率很小^[7~9]。为此,本文针对第2部分中提到的几种情况,提出了一种新的距离集和拐点采样算法——基于边界与采样轮廓一致性的采样算法。实验结果表明,该算法不仅可以很好地适应各种VOP边界走向,并具有良好的编码效率。

2 基准线算法分析

基于基准线的形状编码算法包括拐点的抽取以及距离集采样^[4]。

如图1所示,基准线是指用来计算1维数据(距离集)的基准。基准线选择标准为:若VOP的最小边框矩形宽度大于高度,则基准线为水平轴,反之,基准线为垂直轴。距离集是由形状边界像素点到基准线的距离值构成的集合,而拐点为沿跟踪方向跟踪边界点时,边界走向发生改变后的第1个采样点。如图1中的 S_{12} 为边界走向由向右转变为向左后的第1个采样点,所以 S_{12} 为拐点。同理, S_{14} 、 S_{19} 也是拐点。

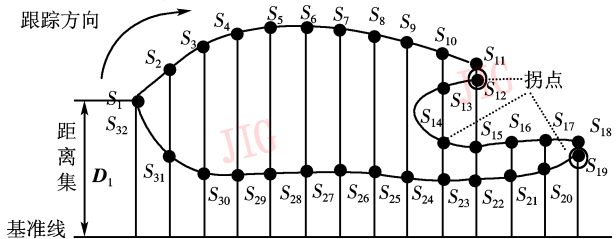


图1 距离集和拐点
Fig. 1 Distance list and turning point

当不对距离集采样时,VOP帧内形状编码称为无损编码。为了进一步提高VOP帧内形状编码的效率,通常以一定的采样率采样距离集,此时的VOP帧内形状编码称为有损编码。采样率通常有1/16、1/8、1/4、1/2、1/15种,其对应于每隔16,8,4,2,1个像素点,对边界进行采样的距离值。无损编码只需对跟踪边界像素起始点、距离集和拐点等进行编码、解码,就可得到VOP形状;而有损编码则在解码过程中,对于解码的两个采样点之间的边界点,通过插值产生。

基准线算法将边界的走向分成如图2所示的4种类型。

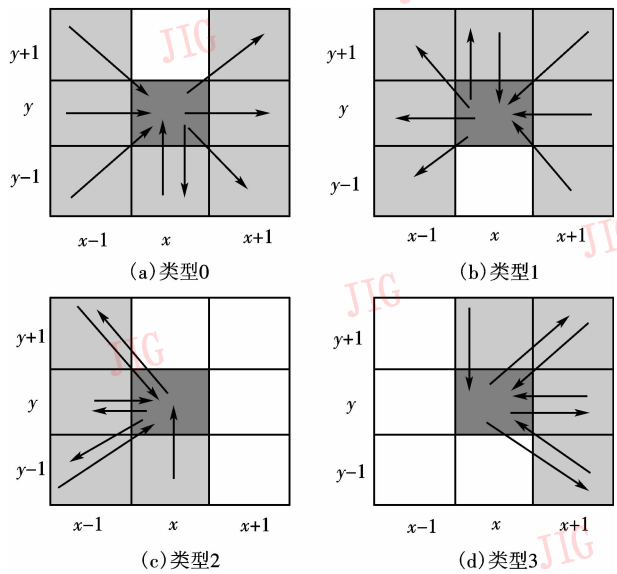


图2 边界走向的4种类型
Fig. 2 Four type of contour pixel

文献[4]基于这4种边界走向类型所设计的距离集和拐点的采样算法步骤如下:

(1) 找到下一个采样点,判断该采样点类型。若该点为类型0,则转步骤(2);若为类型1,则转步骤(3);若为类型2,则转步骤(4);若为类型3,则转步骤(5)。

(2) 抽取该采样点的距离值,如果前一个采样点的类型为类型1,则抽取该采样点为拐点;将前一个采样点类型改为类型0,转至步骤(1)。

(3) 抽取该采样点的距离值,如果前一个采样点的类型为0,则抽取该采样点为拐点;并将前一个采样点类型改为类型1,转至步骤(1)。

(4) 如果前一个采样点的类型为0,则继续执行第4.1步,否则,转至步骤(1);

4.1 如果采样点是距离集上的点,则继续执行第4.2步,否则转到第4.3步;

4.2 对该采样点抽取两次距离值,并抽取一次拐点。转至第4.4步;

4.3 采样点下一个边界点抽取为拐点;

4.4 前一个采样点类型改为类型1;转步骤(1)。

(5) 如果前一个采样点的类型为1,则继续执行第5.1步,否则,转至步骤(1);

5.1 如果采样点是距离集上的点,则继续执行第5.2步,否则转到第5.3步;

5.2 对该采样点抽取两次距离值,并抽取一次拐点,转至第5.4步;

5.3 采样点的下一个边界点抽取为拐点;

5.4 前一个采样点类型改为类型0;转至步骤(1)。

通过对上面的算法进行仔细分析可以发现,对于某些特殊的边界走向,该算法没有提供很好的解决方案。具体分析如下:

情况1 从右向下的边界走向

由于图3(a)中的点A为从右向下的边界走向,但并未包含在4种边界走向类型中,因此,点A将无法被采样,同时也将造成下一个采样点的处理错误。例如,全采样时,图3(a)中的点B应该为拐点,但由于点A无类型(即 $A=0$),因此不会对点B进行采样,也无法得出点B为拐点。而此处拐点错误则会导致解码时的一连串错误。

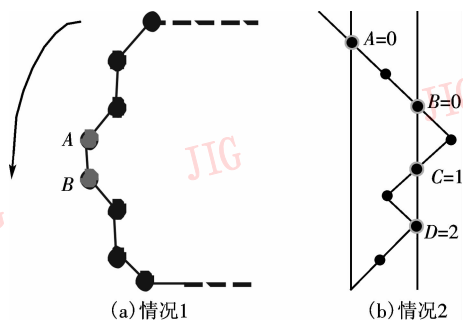


图3 发生错误的情况

Fig. 3 Error cases

情况2 走向转变频繁的边界

图3(b)中如果点A的类型为0(即 $A=0$,下同),点B的类型为0,点C的类型为1(即 $C=1$),点D的类型为2(即 $D=2$),则由点B和点C可得出点C为拐点。但如果点D的类型为2和点C的类

型为1,却无法得出点D为拐点。这样在解码时,解码完点C后,点D的X轴坐标值将减去一个采样间距,而点D的X轴和点C的X轴数值是一样的,结果将出错。

情况3 边界走向和轮廓走向不一致

图4(a)中如果点A的类型为0,点B的类型为2,点C的类型为0,则由点A和点B可得出点B为拐点。但如果点B的类型为2和点C的类型为0,却无法得出点C为拐点。(同理,若点B的类型为3,点C的类型为1,同样无法得出点C为拐点。)

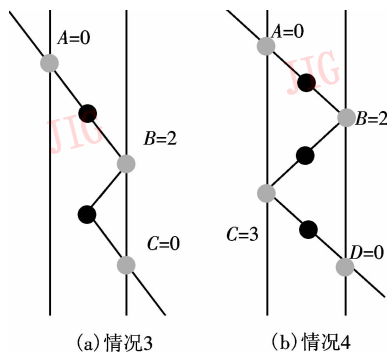


图4 发生错误的情况

Fig. 4 Error cases

情况4 类型2和类型3相邻

如果图4(b)中点A的类型为0,点B的类型为2,点C的类型为3,点D的类型为0,则由点A和点B可得出点B为拐点。但如果点B的类型为2和点C的类型为3,却无法得出点C为拐点。(同理,若点B的类型为3,点C的类型为2,同样无法得出点C为拐点。)

此外,基准线算法的编解码体系也存在一些冗余及考虑不周之处,因此,接下来的第3部分将给出全新的编解码体系。

3 新的基于基准线的形状编码

基准线算法尽管存在多处不足,但利用1维数据表达2维的形状数据的思想还是可取的。为此,本文借用该思想,设计并实现了新的基于基准线的形状编码算法。该算法由新的距离集和拐点采样算法、采样率的确定、帧间编码、编码符号、解码算法等几个部分组成。

3.1 新的距离集和拐点采样算法

新算法将根据边界的实际走向来判断拐点,而

不是根据轮廓采样点的类型来判断拐点。由于边界的方向变化和采样轮廓的方向变化并不完全一致,为此,可将拐点的定义更改如下:边界和采样轮廓的走向都发生改变的前一个采样点定义为拐点(如图 5 所示)。根据新的定义,图 5 中的 S_{11} 为边界和采样轮廓的走向都改变的前一个采样点,因此, S_{11} 为拐点。同理, S_{13} 、 S_{18} 也为拐点。

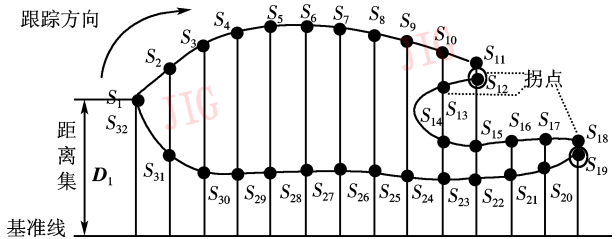


图 5 新算法距离集和拐点的定义

Fig. 5 New turning point and distance list

新算法在遍历边界过程中即可完成距离集和拐点提取。遍历过程中,可根据边界 X (或者 Y) 轴值的增大和减小,将边界走向定义为正方向和负方向。

下面给出新的距离集和拐点采样算法。由于不同基准线方向对算法逻辑没有影响,因此本文接下来的算法都是以 X 轴方向作为基准线方向为例来进行说明。具体算法描述如下:

(1) 遍历边界,找到下一个采样点,如果采样方向为正,则转至步骤(2),否则,转步骤(4)。

(2) 如果当前边界走向依然为正方向,则提取该采样点的距离值并转至步骤(1),否则,继续执行步骤(3)。

(3) 如果前一个拐点减前一个采样点不等于采样间距,则抽取该采样点为拐点,否则,删除前一个拐点;

如果前一个边界点和前一个采样点的 X 值相同,则抽取该采样点的距离值,否则,将前一个采样点 X 值加一个采样距离,并将边界像素指针回溯至前一边界点,转至步骤(1)。

(4) 如果当前边界走向依然为负方向,则提取该采样点的距离值并转至步骤(1),否则,继续执行步骤(5)。

(5) 如果前一个采样点减前一个拐点不等于采样间距,则抽取该采样点为拐点,否则,删除前一个拐点;

如果前一个边界点和前一个采样点的 X 值相同,则抽取该采样点的距离值,否则,将前一个采样

点 X 值减去采样距离。

同时将边界像素指针回溯至前一边界点,转至步骤(1)。

正方向采样时,如果边界走向也是正方向,则将当前采样点的 Y 值放入距离集;否则,算法转入步骤(3)。边界的走向转变了,前一个采样点是否为拐点存在以下两种情况:

(1) 如果前一个拐点和前一个采样点之间的距离不等于采样间距,则说明前一个采样点为拐点,此时需将其放入拐点集;

(2) 如果前一个拐点和前一个采样点之间的距离正好等于采样间距,则说明前一个已提取的拐点和前一个采样点在同一个采样方向上,此时需要将前一个拐点从拐点集中删除。

该算法对于垂直方向上大于两个像素的垂线,若边界走向不改变,则只采样最上面的一个点,而不采样最下面的一个点,而下面一系列点都可在解码过程中得到。若边界走向改变,则采样上、下两个端点,中间的点可在解码时通过插值得到。

3.2 采样率的确定

有损编码过程中,需要选择合适的采样率,采样率的选择办法为:采样率开始取 $1/16$,然后通过线性插值用式(1)计算在相邻两采样点间重构未被采样到的点的距离值。

$$\hat{y}_{k,i} = \frac{(x_{k+1} - \hat{x}_{k,i}) \cdot y_k + (\hat{x}_{k,i} - x_k) \cdot y_{k+1}}{x_{k+1} - x_k} \quad (1)$$

$(x_k \leq \hat{x}_{k,i} \leq x_{k+1})$

式中, y_k, y_{k+1} 分别为与第 $k, k+1$ 个边界采样点的水平坐标 x_k, x_{k+1} 对应的距离值; $\hat{y}_{k,i}$ 为与水平坐标 $\hat{x}_{k,i}$ (第 i 个插值位置)对应的插值距离值。当采用式(2)来计算边界段重构的距离值和原来距离值平均绝对差值和 $\overline{SAD}$ (sum of absolute difference) 时,若 $\overline{SAD} \leq T$ (T 为设定阈值,根据对视频质量要求不同,可以设置不同取值),则采样率取 $1/16$;反之,则采样率取 $1/8$ 。再计算 $\overline{SAD}$,并判断是否 $\overline{SAD} \leq T$,类似采样率取 $1/4, 1/2, 1/1$,直到满足 $\overline{SAD} \leq T$ 为止。式(2)中, $\tilde{y}_{k,i}$ 表示在边界段第 k 部分的第 i 个位置对应边界点的距离值。

$$\overline{SAD}(s) = \frac{1}{16} \sum_{k=0}^{16s-1} \sum_{i=0}^{(1/s)-1} |\hat{y}_{k,i} - \tilde{y}_{k,i}|$$
$$s = \frac{1}{16}, \frac{1}{8}, \dots, 1 \quad (2)$$

3.3 帧间编码

基于基准线的帧间形状编码的目的是为了减少

形状编码在时间上冗余。其主要采用运动估计和补偿技术,即通过前一帧 VOP 形状来预测当前帧的 VOP 形状,并对预测误差进行补偿。

3.3.1 参考形状搜索

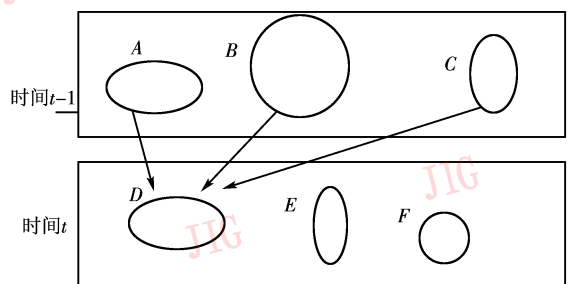
在预测帧中搜索与当前帧的 VOP 形状最匹配形状的过程称为参考形状搜索(如图 6(a)所示),也是 VOP 帧间形状的全局运动估计过程。搜索过程:预测帧的 VOP 形状在一定的搜索范围(一般为 $[-8, +8]$)与当前帧的 VOP 的形状匹配,并按式(3)计算预测帧的 VOP 形状边界和当前帧的 VOP 形状边界之间的匹配误差平均绝对差值和 $\overline{SAD}$,其中最小误差的预测帧的 VOP 形状就为参考形状。

$$\overline{SAD}(m, n) = \frac{1}{L-1} \sum_{l=0}^{L-1} \sum_{k=0}^{N_l-1} \left| Y_{2k,l}(m, n) - Y_{2k+1,l}(m, n) \right| \quad (3)$$

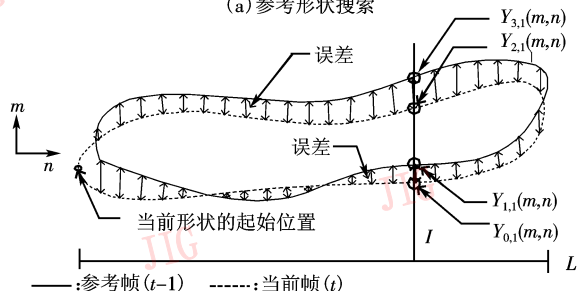
上式中, $Y_{2k,l}(m, n)$ 、 $Y_{2k+1,l}(m, n)$ 分别表示在与基准线相垂直的第 l 个垂直线对应的第 k 对距离值; L 表示基准线的长度; N_l 表示基准线第 l 个位置的对应距离值的配对数; (m, n) 表示搜索位置的坐标。图 6(b) 表示形状边界误差计算。为了提高运动估计的时间效率,可以设置误差容忍阈值 $T_{\overline{SAD}}$, 当 $\overline{SAD}(m, n) \leq T_{\overline{SAD}}$ 时,则认为找到的就是参考形状,并跳出循环。此时的运动矢量就为全局运动矢量。在基于块的视频编码的运动估计中,该方法可能让运动估计陷入局部最优点,但对于形状搜索的全局运动估计而言,由于一个视频帧有完全相同视频对象形状的机率几乎不存在,而且不同视频对象的形状之间的 $\overline{SAD}(m, n)$ 必然较同一视频对象形状之间的 $\overline{SAD}(m, n)$ 有较大差别,因此,一个较合适的 $T_{\overline{SAD}}$ 完全可以得到最佳的运动估计,而不会陷入局部最优点。

3.3.2 边界段的运动估计和补偿

找到参考形状后,预测帧的 VOP 的参考形状可通过全局运动移到(当全局运动较小时,可不移动)与当前帧的 VOP 形状最佳的匹配位置,以便进行边界段的运动估计和补偿。这种运动估计也即是边界段的局部运动估计。边界段运动估计和补偿过程为:参考形状移到最佳匹配位置,每一参考形状边界段在一定的搜索范围(一般为 $[-4, +4]$)与当前帧 VOP 的形状边界段匹配,并按式(4)计算参考形状



(a) 参考形状搜索



(b) 形状边界的误差计算

图 6 参考形状搜索和边界误差计算

Fig. 6 Search reference shape

边界段和当前帧形状边界段的匹配误差平均绝对差值和 $\overline{SAD}$,而与最小误差处对应的参考形状边界段即为边界段的运动估计结果,其对应的边界段距离值误差为补偿值。

$$\overline{SAD}(m, n) = \frac{1}{L-1} \sum_{l=0}^{L-1} \sum_{k=0}^{N_l-1} \left| Y_{2k,x_{\text{pos}}(l)}(m, n) - Y_{2k+1,x_{\text{pos}}(l)}(m, n) \right| \quad (4)$$

式中, $Y_{2k,x_{\text{pos}}(l)}(m, n)$ 、 $Y_{2k+1,x_{\text{pos}}(l)}(m, n)$ 分别表示与基准线垂直的第 $x_{\text{pos}}(l)$ 条直线对应的第 k 对距离值; $x_{\text{pos}}(l)$ 表示与第 l 个跟踪边界点对应的 x 坐标; (m, n) 表示搜索位置的坐标。

3.4 编码符号

对于采样得到的距离集和拐点数据,可采用差分脉冲编码调制(differential pulse code modulation, DPCM)编码。此外,其他的一些解码必需的信息也需要编码。形状解码所需要的数据或编码符号如表 1 所示。

$Baseline_Dir$ 为编码基准线的方向,基准线的方向包括垂直和水平两种情况;此外,一个 VOP 中可能包含多个对象形状,一个形状中也可能包含有孔洞,分别为 $Shape_number$ 和 $Hole_number$;为了解码出每个对象形状,需要知道编码形状的全局运动矢量、轮廓段运动矢量、拐点的数目、拐点的 DPCM 数值、采样率和距离集的 DPCM 数值。

表 1 编码符号列表

Tab. 1 Encoded symbol Lists

符号	注释
<i>Baseline_Dir</i>	基准线的方向
<i>Shape_number</i>	外部形状个数
<i>Hole_number</i>	内部孔洞形状个数
<i>Start_posDiff</i>	每个形状的起始点在基准线方向的数值
<i>Tp_number</i>	每个形状的拐点数目
<i>First_disValue</i>	每个形状起始点的距离值
<i>Shape_V_x^{motion}</i>	全局运动估计的 X 轴值
<i>Shape_V_y^{motion}</i>	全局运动估计的 Y 轴值
<i>Coding_mode</i>	编码模式(帧内/帧间)
<i>Contour_V^{motion}</i>	每个形状的各个边界段的运动估计矢量
<i>Sampling_rate</i>	采样率
<i>Tp_difference</i>	拐点的 DPCM 数值
<i>Dist_difference</i>	距离集的 DPCM 数值

3.5 形状解码

解码端从码流中提取出 1 维数据信息后,即可通过解码得到形状信息,解码算法如下(以基准线方向为 X 轴方向为例):

- (1) 依次遍历距离集中的每一个距离值;
- (2) 如果是正方向解码,则当前解码的点的 X 轴值加上一个采样间隔,否则将当前解码的点的 X 轴值减去一个采样间隔;
- (3) 解码当前采样点;
- (4) 如果两个采样点之间的间距大于 1,则插值产生两个采样点之间的边界点。
- (5) 判断当前采样点是否为拐点,以便决定接

下来的采样方向是否改变。转至步骤(1)。

解码过程为编码的逆过程。算法流程简述如下:遍历距离集,并根据采样间隔和解码方向得到当前解码点的 X 轴值。如果两个采样点之间的间距大于 1,则在当前解码点和前一个解码点进行插值,每产生一个插值点,还需要判断该插值点和前一个插值点之间的垂直落差是否大于 1,若大于 1,则说明存在垂直线,需要插值产生这些点。插值产生完边界段后,还需要判断当前点是否为拐点,若是,则需要改变解码方向,同时对 X 轴值进行加上采样间隔或者减去采样间隔的调整,以方便后面的解码。

4 实验结果

为验证本文算法编码效果,选用如下视频序列:“Bream”、“Foreman”、“Mother and Daughter(MD)” (如图 7 所示)进行了编码实验。图 7 上面一行为各视频系列的第 1 帧,下面一行为提取的对应的二值形状图。在拐点和距离集的采样过程中,由大量的统计实验可知,当设置的平均误差绝对值和的容忍阈值为 $T_{SAD} = 1.0$ 时,可以保持较好的主观解码质量。所以,本文实验中取 $T_{SAD} = 1.0$ 。表 2 给出的是对上述视频系列进行编码的采样率和峰值信噪比(peak ratio of signal to noise,PSNR)值。编码前后的形状对比如图 8 所示,上一幅为编码前提取的形状信息,下一幅为编码后通过 DPCM 解码得到的形状信息。

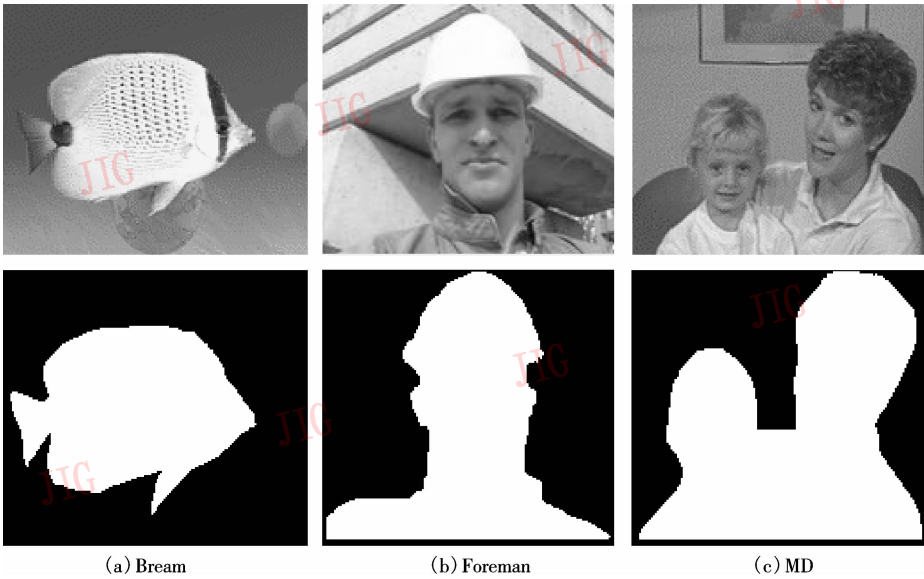


图 7 视频截图和相应的二值形状图片

Fig. 7 Video frame and corresponding binary shape

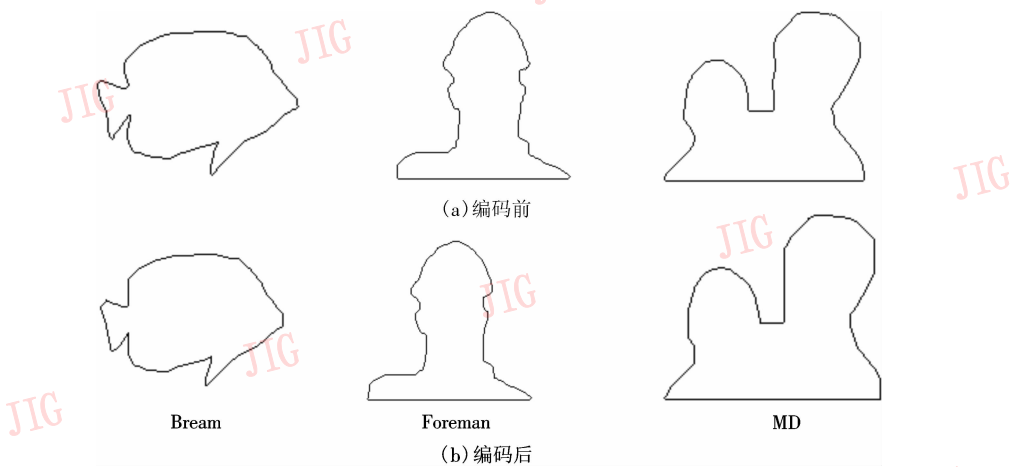


图 8 编解码前后形状对比

Fig. 8 The reconstructed shape boundary

表 2 采样率和 PSNR 值
Tab. 2 Sample rate and PSNR

	Bream	Foreman	MD
$\overline{SAD}$	0.667	0.81	0.728
CR	1/4	1/8	1/4
$PSNR(dB)$	12.63	8.84	8.04

5 结 论

基于基准线的形状编码属于基于轮廓的形状编码方法,其和基于位图的形状编码相比,即使是在有损编码的情况下也能够很好地保持形状的主要特性。本文提出的算法不仅可以很好地适应各种形状,且具有好的编码效率。就算法的时间复杂度而言,基于位图的形状编码的时间花费正比于形状矩阵的分块大小(分块一般为 8×8 或者 16×16),而基于轮廓的形状编码的时间花费则是正比于轮廓大小。大多数情况下,由于形状轮廓要大于形状矩阵的分块,因此,基于轮廓的形状编码的时间复杂度要大于基于位图的形状编码,这也是今后需要进一步研究的工作。

参考文献 (References)

1 Paulo Nunes, Ferran Marques, Fernando Pereira, *et al.* A contour-based

approach to binary shape coding using a multiple grid chain code[J]. Signal Processing: Image Communication, 2000, **15**(7): 585 ~ 599.

2 Chung Jae-won, Lee Jin-hak, Moon Joo-hee, *et al.* A new vertex-based binary shape coder for high coding efficiency [J]. Signal Processing: Image Communication, 2000, **15**(7): 665 ~ 684.

3 Luo Hui-tao. Image-dependent shape coding and representation[J]. IEEE Transactions on Circuits and System for Video Technology, 2005, **15**(3): 345 ~ 354.

4 Lee Shi-hwa, Cho Dar-sung. Binary shape coding using baseline-based method[J]. IEEE Transactions on Circuits and System for Video Technology, 1999, **9**(1): 44 ~ 58.

5 Wang Hao-hong, Schuster Guido M, Katsaggelos A K. An efficient rate-distortion optimal shape coding approach utilizing a skeleton-based decomposition[J]. IEEE Transactions on Image Processing, 2003, **12**(10): 1181 ~ 1193.

6 Chen Zhen-zhong, Ngan King Ngi. Joint texture-shape optimization for MPEG-4 multiple video objects[J]. IEEE Transactions on Circuits and System for Video Technology, 2005, **15**(9): 1170 ~ 1174.

7 Brady Noel, Bossen Frank. Shape compression of moving objects[J]. Signal Processing: Image Communication, 2000, **15**(7): 601 ~ 617.

8 Kim Jong I, Bovik Alen C, Evans Brian L. Generalized predictive binary shape coding using polygon approximation [J]. Signal Processing: Image Communication, 2000, **15**(7): 643 ~ 663.

9 Kadono Shinya, Boon Choong Seng, Ettoh Minoru. Motion compensation method for moving pictures with binary shape [J]. Signal Processing: Image Communication, 2000, **16**(3): 295 ~ 306.