

文章编号:1001-9081(2006)09-2209-02

## 基于 H. 264 码率控制算法的改进

吴东伟,樊养余,赖昌材

(西北工业大学 电子信息学院,陕西 西安 710072)

(wqqq111@sohu.com)

**摘要:**码率控制作为 H. 264 中的重要技术之一,在平稳码率和保证良好的接收端质量方面起着重要的作用。针对 Joint Model (JM) 采用的算法存在的对不同的初始量化参数码率不平稳和 PSNR (峰值信噪比) 差异过大等问题进行分析,在原有算法基础上作出一定的改进。实验结果表明,改进的算法能很好的保证码率和 PSNR 的平稳性。

**关键词:**视频压缩; 码率控制; H. 264 标准

**中图分类号:** TN919 **文献标识码:** A

## Improved rate control algorithm for the H. 264 encoder

WU Dong-wei, FAN Yang-yu, LAI Cang-cai

(Department of Electronic and Information Engineering, Northwestern Polytechnical University, Xi'an Shaanxi 710072, China)

**Abstract:** As one important technology of H. 264, rate control plays an important role in promising a steady rate and a high receiving quality. This article gave a detailed analysis about the problem that the algorithm employed by JM (Joint Model) achieved insecure rate and PSNR (Peak Signal to Noise Ratio) for different preliminary quantization parameters, and made some improvement about the algorithm. The test results demonstrate that the improved algorithm could guarantee steady rate and PSNR.

**Key words:** video compression; rate control; H. 264

### 0 引言

码率控制在所有的视频编码器中都起着重要的作用,虽然目前各种视频编码标准没有对码率控制做明确规定,但是都为自己的编码器推荐了一些提案,例如 MPEG-2 的 TM5<sup>[1]</sup>, H. 263 的 TMN8<sup>[2]</sup>, MPEG-4 的 VM8<sup>[3]</sup> 等。H. 264 码率控制方法的提案主要有两个,一个是马思伟在文献[4,5,6]中提出的基于 MPEG-2/TM5 的改进版本,文中沿用帧比特分配和两次 R-D 模式判别的思想,在宏块级进行码率控制,存在着比特分配不准确以及计算量过大等缺点;另一个是 Li ZhengGuo 在文献[7,8]中提出的基于基本单元的自适应码率控制算法,该算法引入了基本单元,流体流动模型等新的概念,并采用对 MAD 线性预测模型来解决“鸡”和“蛋”悖论,这种算法由于其对图像间相关性的挖掘以及准确的二次 R-D 模型而取得比 F086 提案更好的控制效果。除了上述两种码率控制策略,Simone 在文献[9]中针对量化后零系数所占比率和编码码率之间的近似线性关系提出一种低复杂度的码率控制方法;陈川等在文献[10]中提出一种丢包网络中联合信源信道的码率控制方法;李蕾等在文献[11]使用前后两帧图像直方图的 SAD 作为帧编码复杂度的测量准则,提出一种实时的无需二次编码的比特分配和码率控制和码率控制算法。

JVT-H014 作为目前主流的经典算法,存在着一些缺点<sup>[12]</sup>。首先,算法没有充分利用图像的空间和时间相关性,使预测模型以及比特分配不够准确;其次,没有能根据图像复杂度进行不同大小的基本单元的划分。为此,很多人提出了一些改进方法。例如 WuYuan 等在 JVT-0016<sup>[12]</sup> 中提出一种新的更准确的 R-D 模型, Jiang MinQiang 等在文献[13]中

改进了原算法中对一个 GOP 中对剩余比特平均分配的缺陷, Yi XiaoQuan 等在文献[14]中采用了一种更准确的 MAD 的线性预测方法。

### 1 JVT-H014 算法缺陷

算法采用的二次 R-Q 模型如下:

$$R = MAD \times \left( \frac{X1}{Q_{STEP}} + \frac{X2}{Q_{STEP}^2} \right) \quad (1)$$

这一模型用于基本单元的量化参数计算,其中  $R$  代表编码量化系数所需的码字位数,  $Q_{STEP}$  指基本单元的量化步长,  $MAD$  通过以下线性预测模型进行预测:

$$MAD_{cb} = a_1 \times MAD_{pb} + a_2 \quad (2)$$

其中  $MAD_{cb}$  和  $MAD_{pb}$  代表当前基本单元和前一帧相应位置处的  $MAD$ ,  $X1, X2, a_1, a_2$  是模型系数,在每一个基本单元的最后—一个宏块处理中通过线性回归的方法进行更新。

算法中,第一个 GOP 的  $I$  帧和第一个  $P$  帧采用预先定义的量化参数  $QP_0$  来进行编码,其他 GOP 的量化参数  $QP_{st}$  通过下面的公式计算得到:

$$QP_{st} = \frac{Sum_{PQP}}{N_p} - \frac{8 \times T_r(n_{i-1}, N_{gop})}{T_r(n_{i,0})} - \min \left\{ 2, \frac{N_{gop}}{15} \right\} \quad (3)$$

$N_p$  是前一个 GOP 中  $P$  帧的总数,  $Sum_{PQP}$  是前一个 GOP 中  $P$  帧的量化参数总和。

我们认为,通过上述方法计算的  $QP_0, QP_{st}$  不能充分反应不同场景下的特点,比如在一个运动剧烈和一个几乎没有运动的场景下却可能选用同样的  $QP_0$ ,而下面的实验数据证明同一场景下采用的不同  $QP_0$ , PSNR 差异达到 1dB,而且采用过小的  $QP_0$  可能使码率误差达到 50%,这就说明 JVT-H017

收稿日期:2006-03-28; 修订日期:2006-06-01

作者简介:吴东伟(1981-),男,河南项城人,硕士研究生,主要研究方向:视频压缩; 樊养余(1960-),男,教授,博士生导师,博士,主要研究方向:图像处理、模式识别、虚拟现实; 赖昌材(1977-),江西龙南人,博士研究生,主要研究方向:视频压缩。

的算法没有很好地适应不同的场景。

我们的测试平台是 JM8.4, 对目标比特率为 128kbps, 帧率为 30 f/s 的 100 帧 foreman. qcif 序列和目标比特率为 64kbps, 帧率为 30f/s 的 100 帧 grandma. qcif 序列分别进行测试的结果如下:

表 1 foreman 的测试结果

$QP_0$	PSNR	码率
15	35.09	193.73
16	34.54	172.02
17	34.05	153.33
18	33.50	136.87
19	33.46	128.71
20	33.75	128.34
21	34.21	129.03
22	34.35	128.79
23	34.47	128.60
24	34.44	128.82
25	34.56	128.01
26	34.59	128.61
27	34.63	128.62
28	34.62	128.69
29	34.63	128.94
30	34.65	128.96
31	34.57	128.70
32	34.49	128.80
33	34.48	128.70
34	34.48	128.83

表 2 grandma 的测试结果

$QP_0$	PSNR	码率
20	35.34	72.10
21	34.94	65.52
22	34.84	64.25
23	35.06	64.39
24	35.19	64.42
25	35.23	64.44
26	35.16	64.45
27	35.34	64.41
28	35.13	64.33
29	35.19	64.61
30	35.24	64.42
31	35.20	64.42
32	35.00	64.30
33	35.02	64.36
34	34.91	64.43
35	34.96	64.70
36	34.70	64.31
37	34.83	64.41
38	35.54	64.45
40	34.44	64.41

追究原因,是由于算法没有充分考虑前一 GOP 的码率对当前 GOP 的  $QP_{st}$  的影响,因此采用了不准确的  $QP_{st}$  对 I 帧进行编码,而一个 I 帧要占用比一个 P 帧多达 10 倍的码字<sup>[4~6]</sup>,这点我们可以从 JVT-F086 中对 I 帧和 P 帧初始化分配的码字可见,因此造成了 GOP 内的比特分配不准确,进而使整个序列的码率和压缩质量不平稳。

## 2 改进措施

假设  $B_{i-1}$  代表第  $(i-1)$  GOP 编码产生的码流长度,  $Q_{AStep}$  代表第  $(i-1)$  GOP 中 P 帧的  $QP$  的平均值所对应的量化步长,我们借用(1)来推导更精确的  $QP_{st}$ 。

$$R_{i-1} = MAD \times \left( \frac{X1}{Q_{AStep}} + \frac{X2}{Q_{AStep}^2} \right) \quad (4)$$

我们用  $T_i$  表示第  $i$  个 GOP 分配的目标比特,用  $Q_{STStep}$  代表  $QP_{st}$  对应的量化步长,再次借用(1),

$$T_i = MAD \times \left( \frac{X1}{Q_{STStep}} + \frac{X2}{Q_{STStep}^2} \right) \quad (5)$$

$$\text{令 } \alpha = \frac{R_{i-1}}{T_i(n_{i,0})}, \beta = \frac{X2}{X1 \times Q_{AStep}}, \gamma = \frac{Q_{STStep}}{Q_{AStep}}, \text{将(4), (5)}$$

相除我们可以得到,

$$r^2 - \frac{\alpha}{1+\beta} \times \gamma - \frac{\alpha \times \beta}{1+\beta} = 0 \quad (6)$$

对(6)求解我们可以得到

$$Q_{STStep} = \frac{2}{\sqrt{1 + \frac{4 \times (1+\beta)}{\alpha}}} \times Q_{AStep} \quad (7)$$

利用(7)我们就得到了  $QP_{ST}$ , 设  $QP_{Paverage}$  为  $Q_{AStep}$  对应的量化参数,我们对  $QP_{ST}$  加以限制,

$$QP_{ST} = \min(\max(QP_{Paverage} - 2, QP_{ST}), QP_{Paverage} + 2) \quad (8)$$

$\beta$  初始化为 1,在对序列处理中进行更新。

## 3 实验结果及分析

基于 JM8.4 程序进行实验,实验中仍使用 foreman 和 grandma 序列,采用 CAVLC 编码,基本单元大小为 11 个宏块, GOP 长度为 10。

为了测试改进措施是否能取得更好的效果,我们定义

$$PSNR_{RATIO} = \frac{\max(PSNR - PSNR_{Average})}{PSNR_{Average}} \quad (9)$$

$$RATE_{RATIO} = \frac{\max(R - R_{Average})}{R_{Average}} \quad (10)$$

式中  $PSNR, R$  代表不同  $QP_0$  下取得的  $PSNR$  和码率,  $PSNR_{Average}$  和  $R_{Average}$  代表  $PSNR$  和  $R$  的平均值。

对 foreman 序列在  $19 \leq QP_0 \leq 34$  的情况下,未改进的  $PSNR_{RATIO} = 2.73\%$ ,改进后的  $PSNR_{RATIO} = 1.40\%$ ,在  $15 \leq QP_0 \leq 34$  的情况下,未改进的  $RATE_{RATIO} = 42.7\%$ ,改进后的  $RATE_{RATIO} = 0.23\%$ ;对 grandma 序列在  $20 \leq QP_0 \leq 40$  的情况下,未改进的  $PSNR_{RATIO} = 1.42\%$ ,改进后的  $PSNR_{RATIO} = 1.27\%$ ,未改进的  $RATE_{RATIO} = 11.19\%$ ,改进后的  $RATE_{RATIO} = 0.23\%$ 。并且,两个序列改进后的  $PSNR_{Average}$  均比改进前提高 0.1dB 左右。

为了更直观地改进效果的平稳性,图 1 描绘了两序列在不同的  $QP_0$  下 PSNR 和码率的对比情况。

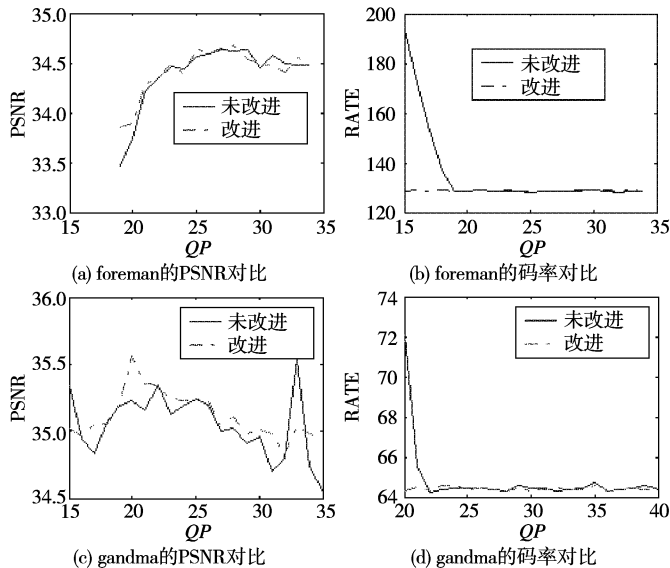


图 1 两序列在不同的  $QP_0$  下 PSNR 和码率的对比

可以看出,改进的方法能很好的改善 JVT-H014 存在的不同  $QP_0$  下码率和 PSNR 不平稳的问题。

## 4 结语

本文通过实验发现 JVT 提案存在的问题,提出整改措施,使改进方法能在不同的应用场景下具有更好的适应性。码率控制的关键在于充分利用图像的空间和时间相关性,更准确地预测码率和失真之间的关系,因此, JVT-H017 提案还有可以改进的地方,比如对基本单元是否能提出一种自适应的改变其大小的措施,对图像复杂度和码率之间的关系能否提出更精确的模型等。

### 参考文献:

- [1] ISO-IEC/JTC1/SC29/WG11/N0400[S]. Test Model 5, 1993.
- [2] ITU-T/SG15[S]. Video Codec Test Model, Near-Team, TMN8. ITU Study Group 16, Video Coding Experts Groups, Document Q15-A-59, Portland, USA, 1997.

(下转第 2224 页)

清单 4:ResultSet 回调模式的客户端代码

```
JdbcUtils ju = new JdbcUtils( getDataSource());
String sql = "SELECT authorId, name FROM author";
List authors = ju.queryForList( sql, new RowMapper() {
    public Object mapRow( ResultSet rs)
        throws SQLException { // 执行一条记录的转换
        Author author = new Author();
        author.setAuthorId( rs.getInt(1));
        author.setName( rs.getString(2));
        return author;
    }
});
```

其中接口 RowMapper 的 mapRow(ResultSet) 方法是具体的业务逻辑,由回调接口根据具体的实现进行处理。清单 5 列出了模板方法 queryForList(String, RowMapper) 的具体处理过程,它本身则实现了 ResultSetCallback 的回调接口:

清单 5:ResultSet 回调模式的具体处理过程

```
public List queryForList( String sql, final RowMapper rowMapper) {
    return (List) query( sql, new ResultSetCallback() {
        // 实现 ResultSetCallback 的回调接口
        public Object doInResultSet( ResultSet rs)
            throws SQLException {
            List results = new ArrayList();
            while ( rs.next() ) {
                // 执行 RowMapper 的回调方法
                results.add( rowMapper.mapRow( rs) );
            }
            return results;
        }
    });
}
```

模板方法 query( String, ResultSetCallback) 的实现与清单 3 比较类似,不再赘述。

### 3 性能差异

testQueryByJU() 函数采用清单 4 的算法实现; testQueryByJDBC() 函数采用清单 1 的算法实现,函数返回值为消耗的时间间隔(ms),执行的功能均将 Author 对象封装到 List 中。查询 5000 条记录。

testQueryByJU() 与 testQueryByJDBC() 分别执行 20 次查询,得出查询总消耗时间  $s1$  和  $s2$  以及平均消耗时间  $t1$  和  $t2$ ; 经过 10 次这样的比较后得出两者总时间消耗的比率  $r$ 。

利用上述的算法,得出的数据和结果如表 1 所示。

表 1 性能比较

序号	$s1/ms$	$s2/ms$	$t1/ms$	$t2/ms$
1	1091.0	1062.0	54.55	53.10
2	1082.0	1001.0	54.10	50.05
3	1021.0	1032.0	51.05	51.60
4	1031.0	1112.0	51.55	55.60
5	1022.0	1011.0	51.10	50.55
6	1092.0	1011.0	54.60	50.55
7	1021.0	1002.0	51.05	50.10
8	1021.0	1042.0	51.05	52.10
9	981.0	1052.0	49.05	52.60
10	1031.0	1012.0	51.55	50.60

$$r = \frac{\sum s1}{\sum s2} \approx 1.0054$$

根据测试的结果看,封装后的函数性能与直接采用 JDBC API 的性能差异不大,大约只有 5.4% 的损失。

### 4 结语

利用模板方法,通过回调用户实现的处理接口,共同完成了数据库数据的存取所需要的功能。可根据实际情况创建一系列默认的回调方式和模板方法对其进行扩展,而针对特殊的应用则可实现回调接口来完成。

采用模板方法与回调函数返回的结果与数据库再无关联。虽然相比直接采用 JDBC API 略有性能上的损失,但数据库资源及时的释放可快速响应其他的处理请求,在并发情况下有助于提高数据库访问的性能。

花费微小的性能代价可获得自动的资源管理、一致的编程模型以及良好的灵活性与扩展性,让程序员集中精力处理业务逻辑,提高了工作效率与质量,使得整个应用更加稳定健壮。此外,由于该方案基于标准的 JDBC API,不依赖于其他任何基础架构,例如 EJB 容器等,程序的调用和调试简单,可单独使用,也可集成到其他框架中作为数据库操作的底层接口。

#### 参考文献:

- [1] ELLIS J, HO L. JDBC 3.0 Specification[EB/OL]. <http://java.sun.com/products/jdbc/download.html>, 2001.
- [2] MARX D. Add Some Spring to Your Oracle JDBC Access [EB/OL]. [http://www.oracle.com/technology/pub/articles/marx\\_spring.html](http://www.oracle.com/technology/pub/articles/marx_spring.html), 2005.
- [3] BROWN S. JSP 编程指南[M]. 北京: 电子工业出版社, 2002.
- [4] GAMMA E, HELM R, JOHNSON R, *et al.* Design Patterns: Elements of Reusable Object-Oriented Software[M]. Boston: Addison-Wesley Professional, 1995.

(上接第 2210 页)

- [3] CHIANG T, ZHANG Y. A new rate control scheme using quadratic rate distortion model[J]. IEEE Trans, 1997, CSVT-7(1): 287 - 311.
- [4] MA SW, GAO W, LU Y. Rate Control on JVT Standard., Document: JVT-D030[A]. JVT4th Meeting[C]. Klagenfurt, Austria, 2002.
- [5] MA SW, GAO W, LU Y, *et al.* Improved Rate Control Algorithm, Document: JVT-E069[A]. JVT5th Meeting[C]. Geneva, CH, 2002.
- [6] MA SW, GAO W, LU Y, *et al.* Proposed draft description of rate control on JVT standard, Document: JVT-F086[A]. JVT6th Meeting[C]. Awaji, 2002.
- [7] LI ZG, PAN F, PANG K. Adaptive Basic Unit Layer Rate Control for JVT, Document: JVT-G012[A]. JVT7th Meeting[C]. Pattaya II, Thailand, 2003.
- [8] LI ZG, GAO W, PAN F. Adaptive Rate Control with HRD Consideration, Document: JVT-H014[A]. JVT8th Meeting[C]. Geneva, 2003.
- [9] MILANI S, CELESTO L, MIAN GA. A Rate Control Algorithm for the H.264 Encoder[EB/OL]. <http://primo.ismb.it/firb/docs/milani1.pdf>.
- [10] 陈川, 余松煜. 联合编码模式选择的码率控制算法[J]. 电子学报, 2004, 32(5).
- [11] 李嵩, 余松煜. 一种高效的 H.264 码率控制方法[J]. 上海交通大学学报, 2004, 38(11).
- [12] YUAN W, LIN SX, ZHANG YD, *et al.* Optimum Bit Allocation and Rate Control for H.264/AVC, Document: JVT-O016[A]. JVT 15th Meeting[C]. Busan, KR, 2005.
- [13] JIANG MQ, YI XQ, LING N. Improved Frame-Layer Rate Control for H.264 Using MADRatio[EB/OL]. <http://ieeexplore.ieee.org/search/wrapper.jsp?arnumber=1328871>.
- [14] YI XQ, LING N. Rate Control Using Enhanced Frame Complexity Measure For H.264 Video[EB/OL]. [http://students.engr.scu.edu/~xyi/publication/SIPS2004\\_RC.pdf](http://students.engr.scu.edu/~xyi/publication/SIPS2004_RC.pdf).