

基于菱形搜索的改进的运动估计算法研究

李 淳, 马力妮

(北京机械工业学院 计算机及自动化系, 北京 100085)

摘 要: 寻找到最优的运动估计算法以提高图像编码效率, 一直是图像编解码技术中研究的重点。在分析菱形算法优点和不足的基础上, 介绍了改进算法中具有代表性的对大模板作出修改的六边形运动估计算法和基于运动向量预测的高级菱形搜索算法, 并通过实验结果对其各自的优点与缺陷进行了具体分析。

关键词: 运动估计; 块匹配; 六边形搜索; 高级菱形搜索

中图分类号: TP301.6

文献标识码: A

文章编号: 1673-629X(2008)11-0117-03

A Study of Improved Motion Estimation Algorithms Based on Diamond Search

LI Chun, MA Li-ni

(Department of Computer and Automation, Beijing Institute of Machinery, Beijing 100085, China)

Abstract: Finding the best motion estimation algorithm to improve the image encoding efficiency is the key technology studied in the image encoding technology field. On a basis of diamond search, introduces HEXBS that modifies the large template and ADS based on prediction vectors, and analyzes advantages and shortcomings of their own through experiments in the end.

Key words: motion estimation; block matching; hexagon-based search; advanced diamond search

0 引 言

运动搜索估计是视频编码器的重要组成部分, 它影响到残差的大小和运动向量的准确性, 从而极大地影响编码的效果和码率, 所以对于编码的结果的效率影响很大, 是编码器耗时最多的模块之一。有测试表明, 在整个的编码过程中, 运动估计与补偿所占时间大约是整个编码过程的 65%^[1]。

目前已出现了很多种类的运动估计算法, 如: 块匹配法、子波变换法、神经网络法、像素匹配法等。其中, 块匹配法运动估计因其算法简单、便于实现等优点得到广泛应用^[2,3]。MPEG4 的运动估计采用的就是宏块内的块匹配算法。

块匹配算法的基本过程为对于需要帧间补偿编码的宏块, 以其中的每一个块为单位, 到已编码成功的参考帧上的对应块上进行匹配。由于运动的存在, 在编码帧和参考帧对应块之间无法达到完全的匹配, 所以

必须在参考帧对应块周围块中进行搜索, 找到和编码块匹配效果最好的块作为参考块, 同时计算两者之间的运动 (d_x, d_y) 作为运动向量。在编码时, 将每个块运动估计得到的对应块之间的残差和运动向量分别编码作为该宏块的编码结果。在块匹配法中, 重点研究块匹配准则及搜索方法。

目前有三种常用的匹配准则:

(1) 最小绝对差准则(MAD):

$$MAD(i, j) = \frac{1}{MN} \sum_{m=1}^M \sum_{n=1}^N |pre(m, n) - cur(m + i, n + j)| \quad (1)$$

式中 i, j 为视差矢量, pre 和 cur 分别为当前帧和参考帧的亮度值, $M \times N$ 为图像的尺寸。如果在某一点 (i, j) 处 MAD 值达到最小, 则该点就为要找的最优匹配点。由于 SAD(求和绝对差) 不含乘法, 运算简单, 通常用 SAD 替代 MAD。

(2) 绝对误差和(SAD, Sum of Absolute Difference) 准则:

$$SAD(i, j) = M \times N \times MAD(i, j) \quad (2)$$

如果在某一点 (i, j) 处 SAD 值达到最小, 则该点就为要找的最优匹配点。

(3) 归一化互相关函数(NCCF, Normalized Cross

收稿日期: 2008-02-19

基金项目: 北京市教育科技发展计划面上项目(KM200511 2320092)

作者简介: 李 淳(1982-), 女, 硕士研究生, 研究方向为流媒体技术; 马力妮, 副教授, CCF 会员, 研究方向为流媒体技术、数字视频编解码技术。

Correlation Function) 准则:

$$NCCF(i, j) = \frac{\sum_{m=1}^M \sum_{n=1}^N pre(m, n) - cur(m+i, n+j)}{[\sum_{m=1}^M \sum_{n=1}^N pre^2(m, n)]^{\frac{1}{2}} [\sum_{m=1}^M \sum_{n=1}^N cur^2(m+i, n+j)]^{\frac{1}{2}}} \quad (3)$$

全搜索属于全局最优的算法,但是运算量相当大,具有很高的复杂性。若采用全搜索,耗时约占整个编码时间的 50% 以上。目前,国内外已提出了许多快速搜索算法,利用局部最优代替全局最优,搜索点数大大减少,从而达到快速运动估计的目的。最广泛使用的是基于不同搜索模板的快速算法,如 TSS(三步搜索)、NTSS(新三步搜索)、4SS(四步搜索)、DS(钻石搜索)等。

1 传统菱形搜索算法

根据实验数据统计表明,利用全搜索算法计算获得的运动向量分布概率和距离搜索中心点的距离之间的关系可以看出,50%~90% 的运动向量集中在以搜索中心为圆心的半径为 2 的圆上,如图 1 所示。

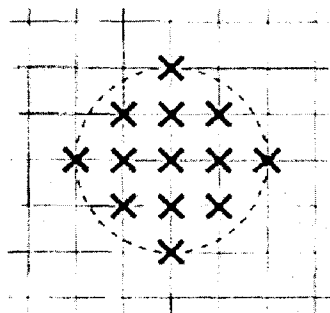


图 1 运动向量主要分布

根据实际视频序列物体运动的统计,实际视频中块的运动可以在任何方向上进行运动,但主要集中在水平和垂直两个方向上(摄像机运动)。所以上图中在半径为 2 的圆中的 13 个搜索点是具有最优匹配概率最大的点。所以在该圆形区域内进行搜索,搜索匹配的点数最小而能获得最佳的搜索效果。基于上述理论,菱形搜索算法被提出。

为了使得搜索范围为以搜索中心为原点的圆,菱形搜索算法采用了两个搜索模式,如图 2 所示。

一个模式称为大菱形搜索模式(LDSP),采用 9 个搜索点,包括搜索中心,以及 8 个按照菱形分布的围绕点。第二个模式成为小菱形搜索模式(SDSP),采用搜索中心和与其相邻的水平垂直方向上的 4 个点共 5 个点组成小菱形。

DS 搜索算法的搜索过程首先以搜索中心为中心,进行大菱形搜索,计算 9 个点,如果最小 MAD 的点不在大菱形的中心的话,将大菱形中心移到相应最小 MAD 的点上,重复大菱形搜索,直到最小 MAD 的点位于大菱形的中心为止。然后在该中心点上切换到小菱形搜索,共搜索 5 个点得到最终的搜索结果作为运动估计的最优匹配点。

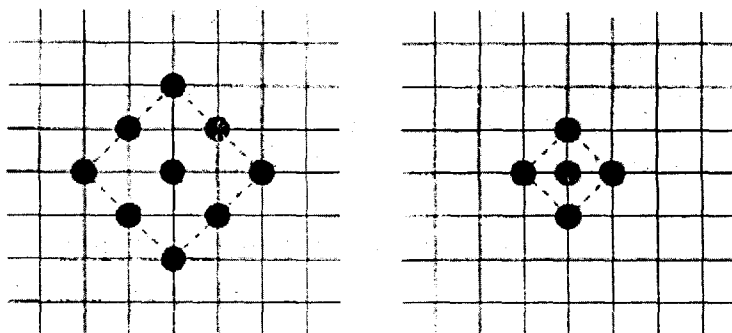


图 2 LDSP 模板与 SDSP 模板

2 改进的运动搜索算法

菱形搜索算法在 1999 年 10 月被 MPEG4 标准采纳并收入验证模型,是通过局部搜索力图达到全局最优的一个比较成功的算法。虽然它的综合性能较其它算法优越,但平均搜索点数仍在 15.5 左右^[4]。如何在尽量不影响其它性能的前提下进一步减少搜索点数,降低运算量,一直是运动估计算法研究的重要目标。

目前提出的运动搜索算法有很多种,最具代表性的有对运动模板作出修改的算法和基于运动向量预测的算法。

2.1 六边形搜索算法 (HEXBS, Hexagon - Based Search)

进一步分析 LDSP 可以发现,LDSP 四周的 8 个匹配点到中心点的距离是不同的,因此使用 LDSP 进行粗定位时,沿不同方向移动的匹配速度也不同,当 LDSP 的顶点为本次匹配的 MBD 点时模板沿水平或垂直方向移动,此时的搜索速度为 2 像素/步;当模板沿对角方向移动时其速度为 $\sqrt{2}$ 像素/步。另一方面,在大模板移动的每一步中,不同的搜索方向需要检测的搜索点数也不同。水平和垂直方向上需要检测 5 个新搜索点,而对角方向上只需检测 3 个新的搜索点即可。从以上几点可以看出,LDSP 模板并不是最优的搜索模板。事实上,造成该问题的根源在于块匹配误差实际上是在搜索范围内建立的误差表面函数,全局最小点即对应着最佳运动矢量,而 LDSP 实际上只是一个旋转了 45° 的正方形模板,在对角方向上的梯度下降方向不过快,需要较多步才能够搜索到最优点^[5]。

六边形搜索算法在菱形搜索的基础上进一步改进。根据搜索模式符合以 2 为半径的圆形使得搜索点数最优的理论,HEXBS 将 DS 的 LDSP 修改成六边形模式,同时 SDSP 仍然保留。如图 3 所示。

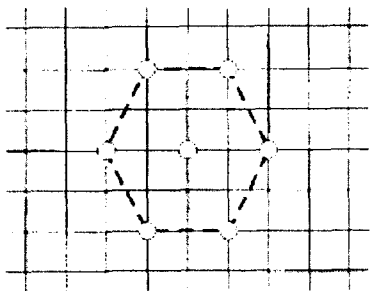


图 3 HEXBS 的 LDSP 模板

这样的改进有两个优点:一个是大的搜索模式更贴近于以 2 为半径的圆,搜索效率更高;另一个是很明显的改进,HEXSP 比 LDSP 减少 2 个搜索点,这样平均搜索点数会至少减少 2 个点^[6]。

2.2 高级菱形搜索算法 (ADS, Advanced Diamond Search)

菱形算法另外一个缺陷就是不能根据图像的内容作出灵活处理,即不管是什么样的运动,一律先用 LDSP 来搜索,再用 SDSP 搜索,这对小运动图像是一种浪费。高级菱形搜索法抛弃了大模板,依据先前的运动向量计算出一个预测向量,然后在预测向量附近进行搜索。算法具体描述为:

Step1:以当前点为中心计算小菱形的 5 个匹配点,如果最小 SAD 点不在中心点则跳转到 Step2,否则跳转到 Step5。

Step2:记录更好的方向;

记录 SAD 最小的点为当前中心点;

如果更好的方向是左右方向,那么测试该位置的上下方向;

如果更好的方向是上下方向,那么测试该位置的左右方向;

如果这次又找到了更好的方向,将更好的方向累加,记录 SAD 最小的点为当前中心点;跳转到 Step3。

Step3:进一步搜索:

如果搜索方向是趋向右边的,那么搜索当前中心点的右上点和右下点;

否则如果搜索方向是趋向左边的,那么搜索当前中心点的左上点和左下点;

否则如果搜索方向是趋向右上的,那么搜索当前中心点的左上点、右上点和右下点;

否则如果搜索方向是趋向上边的,那么搜索当前中心点的左上点和右上点;

否则如果搜索方向是趋向下边的,那么搜索当前中心点的左下点和右下点;

否则搜索方向是趋向左上的,那么搜索当前中心点的左下点、左上点和右上点;

否则如果表明搜索方向是趋向右下的,那么搜索当前中心点的左下点、左上点和右上点;

否则如果搜索方向是趋向左下的,那么搜索当前中心点的左上点、左下点和右下点;

否则就认为本轮搜索没有找到更好的点,那么搜索当前中心点的左上点、左下点、右上点、右下点;

搜索完毕后转到 Step4。

Step4:如果没有找到更好的方向,跳转到 Step5。

否则更新 bDirection 为更好的方向,记录 SAD 最小的点为当前中心点,返回 Step1。

Step6: 停止搜索。

如图 4 所示,每次找到的最佳匹配点加重表示,本算法在第四步搜索结束。由于每轮搜索的点数相应的减少,本算法与 DS 算法相比降低了搜索时间。

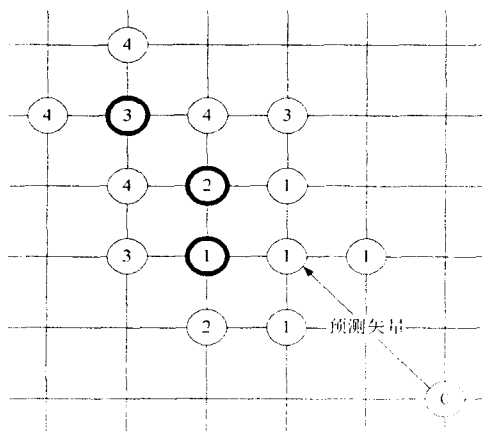


图 4 ADS 搜索示意图

3 算法优劣分析

为了比较算法的性能,分别选取了低空间细节且运动缓慢的 M&D 序列、中等空间细节且运动缓慢的 Foreman 序列、高空间细节且运动剧烈的 Stefan 序列进行了实验。实验采用改进的 xvidcore1.1.2 编码器,使用宏块大小为 8×8 。实验结果如表 1 所示(时间的单位为:ms,PSNR 的单位为:dB)

表 1 实验结果

序列	DS		HEXBS		ADS	
	时间	PSNR	时间	PSNR	时间	PSNR
M&D(qcif)	3.60	39.19	3.49	38.70	2.29	40.02
Foreman(qcif)	3.34	42.84	3.01	41.98	1.93	43.34
Stefan(cif)	15.28	43.22	12.06	43.11	10.24	41.81

通过实验可以发现,HEXBS 算法比 DS 算法在大

(下转第 122 页)

```
//代理的创建
}

public LogHandler (Object target){
    this.target = target;
}

public Object invoke(
    Object proxy,
    Method method, Object[] args)
    throws Throwable {
    //对方法的拦截,通知的添加,都是在该函数中
    Object rs = null;
    try{
        System.out.println("Before calling the operateA method");
        //前置通知
        rs = method.invoke(target,args);
        //调用实际对象的方法
    }
    catch(InvocationTargetException ex){
        System.out.println(ex);
        //异常通知
    }
    System.out.println("Finishing calling the operateA method");
    //后置通知
    return rs;
}
}
```

由以上的实现代码可以看出,动态代理中的接口 `InvocationHandler` 实现是最关键的一步, `InvocationHandler` 接口只定义了一个方法 `invoke` 方法, AOP 的具体实现就体现在该方法的定义之中。切入点和通知的添加主要在 `method.invoke` 函数的前后, 函数中的代

码 `System.out.println("Before calling the operateA method")`; 就是 AOP 中的前置通知; `System.out.println("Finishing calling the OperateA method")`; 是后置通知; 对方法 `method.invoke` 的异常捕捉就是异常通知。在实际应用中需要一个管理切入点和通知的容器, 提供一个 XML 文件配置需要代理的类, 容器根据配置文件动态实现切入点和通知的装配。

3 结束语

以上介绍了通过动态代理的方式实现 AOP 的原理和方法, 除此之外实现 AOP 还有一些其他的方法, 如动态字节码生成、编译期织入和类加载织入, 这些都需要特殊的工具来协助实现。AOP 是一种比较新的技术, 可以很大程度地提高代码的复用, 但 AOP 不可能代替 OOP, 它是 OOP 的一种补充和完善。AOP 能提高效率和复用, 便于维护, 因此 AOP 的应用也会越来越广泛, 现在的主流 J2EE 产品都提出了对 AOP 的支持, 相信在未来的应用中, 会逐渐采用这种技术。

参考文献:

- [1] Walls C, Breidenbach R. Spring In Action[M]. Beijing: Manning Publications, 2007.
- [2] 李 刚. Spring 2.0 宝典[M]. 北京: 电子工业出版社, 2006.
- [3] 王申源, 董传良, 刘英丹. 面向方面的编程的研究与实现[J]. 计算机应用研究, 2004(11): 220-223.
- [4] 石丹丹. 面向方面编程模式的探讨[J]. 武汉理工大学学报, 2005, 27(1): 93-95.
- [5] 张广红, 陈 平. 关于 AOP 实现机制和应用的研究[J]. 计算机工程与设计, 2003(8): 14-17.

(上接第 119 页)

运动情况下更具有优势, 能够更快地搜索到大运动向量的匹配点, 但是有时会导致冗余甚至错误, 这是由于采用的搜索点分布过稀所造成的; 而 ADS 算法由于摒弃了大模板, 所以更适合小运动图像, 对于运动比较激烈的图像, 编码后图像的质量不是很理想。

4 结束语

随着网络和数字化时代的飞速发展, 对于视频图像多媒体的处理要求越来越高, 追求高压缩比、高传输效率、高清晰图像质量是视频技术发展的方向。运动估计对视频编码器是否能实现实时编码和编码的视频质量有着重要影响。因此, 有必要对运动估计做深入研究, 在传统运动估计算法基础上寻求改进方法, 寻找到运动估计速度与精度的平衡点, 以提高编码效率。

参考文献:

- [1] Zhu S, Ma K K. A New Diamond Search Algorithm for Fast Block-matching Motion Estimation[J]. IEEE Trans Image Processing, 2000, 9(2): 287-290.
- [2] 刘 翔. 基于光流场算法的运动估计方法研究[J]. 无线电工程, 2006, 36(4): 17-20.
- [3] 姚 晨, 沙济彰. 用于块匹配运动估计的 SGDS 算法[J]. 计算机与现代化, 2006(1): 8-12.
- [4] 莫路锋, 熬 山, 顾洁宇. 一种基于 MPEG4 的运动估计模板选择算法[J]. 传感技术学报, 2006, 19(3): 36-39.
- [5] 倪 伟, 郭宝龙, 丁贵广. 基于六边形的运动矢量场自适应搜索算法[J]. 计算机工程, 2005, 31(13): 10-12.
- [6] Zhu Ce, Lin Xiao, Chau Lap-Pui. Hexagon-Based Search Pattern for Fast Block Motion Estimation[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2002, 12(5): 349-355.