电子科技大学

硕士学位论文

MPEG4运动估计新算法及工程研究

姓名：欧阳国胜

申请学位级别：硕士

专业：信号与信息处理

指导教师：罗永伦

20070601

摘 要

基于块匹配的运动估计技术对许多视频编码运动补偿技术极其重要，如 ISO 的 MPEG-1/2/4 和 ITU-T 的 H.26x,基于块匹配的运动估计是利用连续序列中时间冗余特性。其基本思想是将当前帧分割成大小相同、不重叠的矩形块或宏块；对于每个块在参考帧的匹配窗内按照一定的匹配准则搜索与之最接近的块。该预测块与到当前块的位移就是运动矢量，预测块和当前块之间的差值称为残差，每个块都可以使用残差块和运动矢量表示。预测越准确，意味着残差中的数值越小，编码所占用的比特数也越少。然而，运动估计的计算量相当大，如果用全搜索法（FS）全部估计搜索窗内所有候选块，那么要占用编码器 80%的运算量。因此极其需要在不牺牲精度的同时加快运动估计过程的快速算法。所以开发出许多有效的快速算法，其中典型的有三步搜索法（TSS），新三步搜索法（NTSS），四步搜索法（4SS），基于块的梯度下降法（BBGDS），菱形搜索法（DS）等。在 TSS, NTSS, 4SS 和 BBGDS 算法中利用了不同步长的正方形搜索模式。而菱形搜索法采用了菱形搜索模式，菱形搜索法和 TSS，NTSS 和 4SS 有着近似的匹配误差但是搜索速度更快。在运动估计中，不同形状不同步长的搜索模式和运动矢量分布的中心偏置特性对搜索速度和性能有着重大的影响。

本论文针对 MPEG4 视频编码运动估计中降低搜索运算量和提高搜索精度这对相互关联的矛盾，进行了算法上的研究与改进，提出了三种新型算法：一种是基于经典菱形算法的趋势菱形算法(TDS)和一种形状自适应算法（SAS），用于整像素快速运动估计；另一种是新型趋势半像素算法(THS)，用于半像素快速运动估计，并成功地运用于处理不同运动复杂度的视频，相应的实验数据显示，这三种新算法比经典菱形和全搜索算法搜索速度更快、搜索点数大幅度减小、重建图像峰值信噪比无明显下降的效果。新算法具有的这些优点，在图像编码方面有较大的应用价值。

另一方面，论文针对 VPROVE 视频分析工具不支持对隔行扫描的 MPEG4 码流解码和分析的原有缺陷，基于对 MPEG4 码流解码方面的分析研究，设计出 Video Analyzer，实现了对隔行扫描的 MPEG4 码流进行解码和分析的功能，实现了 VPROVE 视频分析工具的功能升级，极大地提升了该工具的应用价值。

关键词：运动估计，趋势菱形算法，趋势半像素算法，形状自适应算法

ABSTRACT

Block-matching motion estimation is very important to many motion-compensated video-coding techniques/standards, such as ISO MPEG-1/2/4 and ITU-T H.26x, and is aimed at exploiting the strong temporal redundancy between successive frames. By partitioning a current frame into nonoverlapping rectangular blocks/macroblocks of equal size, a block-matching method attempts to find a block from a reference frame (past or future frame) that best matches a predefined block in the current frame. Matching is performed by minimizing a matching criterion, which in most cases is the mean absolute error/difference between this pair of blocks. The block in the reference moves inside a search window centered arounds the position of the block in the current frame. The best-matched block producing the minimum distortion is searched within the search window in the reference frame. The displacement of the current block with respect to the best-matched reference blocks in the x and y directions compose the motion vector assigned to this current block. However, the motion estimation is quite computationally intensive and can consume up to 80% of the computational power of the encoder if the full search (FS) is used by exhaustively evaluating all possible candidate blocks within the search window. Therefore, fast algorithms are highly desired to significantly speed up the process without sacrificing the distortion seriously. Many computationally efficient variants were developed, among which are typically the three-step search (TSS), new three-step search (NTSS), four-step search (4SS), block-based gradient descent search (BBGDS), and diamond search (DS) algorithms. In TSS, NTSS, 4SS, and BBGDS algorithms, square-shaped search patterns of different sizes are employed. On the other hand, the DS algorithm adopts a diamond-shaped search pattern, which has demonstrated faster processing with similar distortion in comparison with TSS, NTSS and 4SS. In block motion estimation, search patterns with different shapes or sizes and the center-biased characteristics of motion-vector distribution have a large impact on the searching speed and quality of performance. Therefore, some novel algorithms were proposed. In this thesis, two new fast search algorithms are proposed, based on extensive study of the

contradiction of reduction of complexity and improvement of precision in the process of search of MPEG4 coding. The first algorithm is trend-diamond-search algorithm (TDS) based on the improvement of diamond search. TDS is designed for fast full-pixel motion estimation. The second algorithm is trend-halfpixel-search algorithm (or THS), which is designed for fast half-pixel motion estimation. These two algorithms have been successfully applied in dealing with videos of various complexity motions. Experiment data obtained by the proposed algorithms shows very close image quality, less search time and fewer search points compared with that obtained by conventional diamond and full search algorithm. This shows that these two new algorithms are correct and have great advantages in image coding engineering application.

Another work carried out in this thesis is the design of video analyzation tool and improving function of VPROVE in supporting decode and analyse interlace bitstream of MPEG4, based on the study and analyse of the MPEG4 bitstream decoding and intrinsic flaws of VPROVE. This work promotes the value of video analyzation tool and its applications in engineering greatly.

Keywords: motion estimation, trend diamond search algorithm, trend half-pixel search algorithm，shape adaptive search algorith

独 创 性 声 明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得电子科技大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

签名：欧阳国胜　　日期：2007年6月11日

关于论文使用授权的说明

本学位论文作者完全了解电子科技大学有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人授权电子科技大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后应遵守此规定）

签名：欧阳国胜　　导师签名：

日期：2007年6月11日

第一章 概述

1.1 研究工作的背景

数字视频技术在通信和广播领域获得了日益广泛的应用，特别是 20 世纪 90 年代以来，随着互联网和移动通信的迅猛发展，视频信息和多媒体信息在互联网和移动网络中的处理和传输，已经成为热点技术。

众所周知，视频信息具有一系列优点，如直观性、确切性、高效性等等。但是，数字化后的视频和音频信息具有数据海量性，它给信息的存储和传输造成较大的困难，成为阻碍人类有效地获取和使用信息的瓶颈问题之一。因此必须首先解决视频海量数据的压缩编码问题，其次解决压缩后视频质量保证的问题。压缩编码与视频质量是相互矛盾的[1]，是矛盾的两个方面。

研究的目标是：既要有较大的压缩比，又要保证一定的视频质量。因此，研究和开发新型有效的多媒体数据压缩编码方法，以压缩的形式存储和传输这些数据将是最好的选择。为此，人们付出了巨大的辛勤劳动，并已取得丰硕的成果。从 1984 年国际电报电话咨询委员会 CCITT（Consultative Committee on International Telegraphs and Telephones）公布第一个视频编码国际标准以来，至今已有 20 多年了。ITU-T 等国际标准化组织陆续颁布了接近十个视频编码国际标准，大大推动了视频通信和数字电视广播的发展。但视频通信和数字电视广播这两大领域至今的发展仍不能令人满意，主要原因是视频压缩与图像质量之间的矛盾未能很好地解决。

图像编码方法一般分为两代：第一代是基于数据统计，去掉的是数据冗余，称为低层压缩编码方法，如 H.120、H.261、H.263、JPEG、MPEG1、MPEG2；第二代是基于内容，去掉的是内容冗余，如 MPEG4，第二代图像编码方法包括了基于对象（Object-Base）方法称（即中层压缩编码方法）和基于语义（Syntax-Based）方法（即高层压缩编码方法）。基于内容压缩编码方法代表新一代的压缩方法，对其的研究是当前最活跃的研究领域，最早由 Forchheimer（1983 年）提出的，随后 Harashima 等人也展示了不少研究成果。

下面对视频编码发展进行简要回顾。

- **H.120、H.261 和 H.263 的发展过程**

1984 年国际电报电话咨询委员会 CCITT 第 15 研究组发布了数字基群电视会议编码标准 H.120 建议。

1990 年 10 月，CCITT 通过了用于视频会议、可视电话的 H.261 标准，称为“p×64kbit/s 视听业务的视频编解码器”，其中 p 的范围是 1～30，覆盖了整个窄带 ISDN 基群信道速率。H.261 标准的制定和颁布为各种视频通信产品的国际间互通提供了保证，可以认为是视频压缩编码技术发展的一个里程碑，是过去几十年来视频编码技术研究成果的结晶。在此后的一系列国际标准中，如 H.26X 和 MPEGX，都采用了 H.261 的编码算法框架：基于运动补偿的帧间预测和 DCT 变换相结合的混合编码方法，即通过帧间预测减少时域冗余，通过 DCT 变换减少空域冗余。这种混合编码方法具有压缩比高、算法复杂度低的优点，在视频通信中得到广泛的应用。

1996 年 5 月，国际电信联盟电报电话通信部门 ITU-T（International Telecommunications Union for Telegraphs and Telephones Sector）正式颁布 H.263 国际标准，称为“低比特率通信视频编码”。它与 H.261 标准相比，提高了编码效率，能保证在可接受的视频质量前提下允许比 p×64kbit/s 更低的信道码率。之后在 1998 年 1 月 ITU-T 推出了 H.263 标准的第二版本 H.263+，它提供了 12 个新的可协商模式和其它特征，进一步提高了压缩编码性能。如 H.263+允许使用更多的源格式，可扩展性得到改进，它允许多显示率、多速率及多分辨率，增强了视频信息在易误码、易丢包异构网络环境下的传输，增强了应用的灵活性。在 2000 年 11 月又推出了第三版本，称为 H.263++，新增加了 3 个高级模式。H.263 标准的升级，使其应用范围进一步扩大，压缩效率、抗误码能力以及重建图像的主观质量等均得到了提高。

- **JPEG 和 MPEG 的发展过程**

1986 年由国际电报电话咨询委员会（CCITT）和国际标准化协会（ISO）联合组成的联合图像专家组 JPEG（Joint Photographic Expert Group），一直致力于标准化工作，开发研制出静止图像的数字图像压缩编码方法，并于 1992 年 7 月通过了 JPEG 标准。JPEG 标准是一个适用范围广泛的通用标准。它不仅适于静止图像的压缩，而且电视图像序列的帧内图像的压缩编码，也常采用 JPEG 压缩标准。

1988 年 ISO/IEC 信息技术联合委员会成立了运动图像专家组 MPEG（Moving Picture Expert Group）。MPEG 的目标是针对活动图像的数据压缩，但是活动图像与静止图像之间有密切关系，一个视频序列图像，可以看作为独立编码的静止图像序列，只是依照视频速率顺序地显示。MPEG 专家小组的研究内容，不仅仅限

制于数字视频压缩，音频及音频和视频的同步问题都不能脱离视频压缩独立进行。MPEG 专家小组，承担制定了可用于数字存储介质上的视频及相关联音频的国际标准，这些国际标准，简称为 MPEG 系列标准。

1992 年 11 月 MPEG 正式公布了 MPEG1 视频编码标准。MPEG1 是面向比特率大约为 1.5Mb/s 的视频信号的压缩和每通道速率为 64Kb/s、或 128Kb/s、或 192Kb/s 的数据音频信号的压缩。所以，MPEG1 是将数字视频信号和与其相伴随的音频信号在一个可以接受的质量下，能被压缩到比特率约 1.5Mb/s 的一个 MPEG 单一比特流。MPEG1 主要应用于家用 VCD 的视频压缩。为了提高 MPEG1 压缩比，帧内图像数据压缩和帧间图像数据压缩技术必须同时使用。帧内压缩算法与 JPEG 压缩算法大致相同，采用基于 DCT 的变换编码技术，用以减小空域冗余信息；帧间压缩算法，采用预测法和插值法，预测法有因果预测器（纯粹的预测编码），和非因果预测即插值编码。预测误差可再通过 DCT 变换编码处理，进一步压缩。帧间编码技术可减小时间轴方向的冗余信息。

1994 年 11 月 MPEG 公布了 MPEG2 标准，用于数字视频广播（DVB）、家用 DVD 的视频压缩及高清晰度电视（HDTV）。码率从 4Mbit/s、15Mbit/s……直至 100Mbit/s 分别用于不同档次和不同级别的视频压缩中。MPEG2 视频体系要求必须保证与 MPEG1 视频体系向下兼容，并力求满足数字在存储媒体、会议电视、可视电话、数字电视、高清晰度电视（HDTV）、广播、通讯、网络等应用领域中，对多媒体视频、音频通用编码方法日益增长的新需求。压缩编码方法也要求从简单到复杂有不同等级。MPEG2 国际标准详述了数字存储媒体和数字视频通信中的图像信息的编码描述和解码过程。MPEG2 标准支持固定比特率传送、可变比特率传送、随机访问、分级解码、比特流编辑以及一下特殊功能，如：快进播放、快退播放、慢动作、暂停和画面凝固等。

上述六种压缩编码国际标准主要采用了第一代压缩编码方法，如预测编码、变换编码、熵编码及运动补偿。

■ MPEG4 的发展过程

1999 年 12 月份，ISO/IEC 通过了“视听对象的编码标准” ——MPEG4，它除了定义视频压缩编码标准外，还强调了多媒体通信的交互性和灵活性。MPEG 组织于 1999 年 1 月正式公布了 MPEG4 (ISO/IEC 14496) V1.0 版本，1999 年 12 月又公布了 MPEG4 V2.0 版本。MPEG4 的制定初衷主要针对视频会议、可视电话超低比特率压缩编码的需求，在制定过程中，MPEG 组织深深感受到人们对媒体信息，特别是对视频信息的需求由播放型转向基于内容的访问、检索和操作。MPEG4

与前面提到的 JPEG、MPEG1、MPEG2 有很大的不同，它为多媒体数据压缩编码提供了更为广阔的平台，它定义的是一种格式一种框架，而不是具体算法，它希望建立一种更自由的通信与开发环境。于是 MPEG4 新的目标就定义为：支持多种多媒体的应用，特别是多媒体信息基于内容的检索和访问，可根据不同的应用需要，现场配置解码器。编码系统也是开放的，可随时加入新的有效的算法模块，为了支持对视频内容的访问，MPEG4 引入了视频对象(Video Object)的概念，所谓对象就是在一个场景中能够访问（搜索和浏览）和操作（剪贴）的实体，MPEG4 采用了基于对象的压缩编码方法，它把图像和视频分成不同的对象，分别处理，基于对象的编码除了能提高数据压缩比，还能实现许多基于内容的交换功能，它可以广泛地用在基于对象地对象的多媒体存取、网上购物和电子商店、远程监控、医疗和教学等。

1.2 原有方法的优点和缺点

视频序列图像在时间上存在很强的相关性，采用运动估计和运动补偿技术可以消除时间冗余以提高编码效率，这种技术已广泛用于视频压缩的一些国际标准中，如 ITU-T H.261/263/264、MPEG1/2/4。在这些视频压缩国际标准中视频系统编码器的效率主要取决于运动估计算法，而运动估计的效率主要体现在图像质量、压缩码率和搜索速度 3 个方面。而图像质量、压缩码率和搜索速度又是由所采用的搜索策略、匹配准则和初始搜索点的选择等因素决定的。

基于块的运动估计是视频压缩国际标准中广泛采用的关键技术。作为一种降低视频序列帧间时间冗余度的有效手段，运动估计在现今的视频编码和处理系统中扮演着重要的角色。出于简单性和编码有效性的考虑，块匹配算法[2]（BMA）已经成为了许多基于运动补偿的视频编码标准中至关重要的部分，例如：ISO MPEG1/2/4 和 ITU-T H.261/263/264。块匹配算法首先将当前帧分割成互不重叠的相同尺寸的矩形块，并在参考帧中定义搜索区域，然后当前帧的每个块与搜索区域中的候选块作比较。通过在由块失真准则（BDM）函数组成的误差曲面上搜索最小点来获得最终的运动矢量。

在现有的各类快速运动估计方法中，块匹配运动估计算法因具有算法简单、便于 VLSI 实现等优点得到广泛应用。目前，搜索精度最高的是全搜索法（FS），它对搜索范围内的每一个像素点进行匹配运算以得到一个最优的运动矢量。但它的计算量巨大，在一些要求运算时间很短的实时场合往往不能达到很令人满意的

效果。为此提出了各种快速运动估计算法，如 1981 年 Koga T、Linuma K、Hirano A 提出的三步法[3]（TSS）、1981 年 Jain J R 和 Jain A K 提出的对数搜索法[4]（LOGS）、1990 年 Ghanbari M 提出的交叉法[5]（CS），主要是通过限制搜索位置的数目来减少计算量。但它们在第一步中搜索步长较大，不利于估计小的运动块。1993 年 Lee W 和 Wang J Y 提出的动态搜索窗调整法[6]（DSWA）根据当前结果动态调整下一步搜索步长的大小，算法性能一定程度上有了改进。1994 年 Li R、Zeng B、Liou M L 提出的新三步法[7]（NTSS）、1996 年 Po L M 和 Ma W C 提出新四步法[8]（NFSS）、1996 年 Liu L k 和 Feig E 提出的基于块的剃度下降法[9]（BBGDS）等利用运动矢量具有中心偏移的分布特性，提高了匹配速度，减少了陷入局部极小的可能性，但它们都是以原点作为初始搜索中心，没有充分利用相邻块之间的运动相关性。1997 年 Luo L J、Zou C R、Gao X Q 提出的预测搜索法[10]（PSA），1999 年 Xu J B、Po L M、Cheng C K 提出的自适应运动跟踪法[11]（AMTS）等利用相邻块的运动相关性选择一个反应当前运动块运动趋势的预测点作为初始搜索点，以提高搜索速度和预测的准确性。1997 年 Shan Zhu 和 Kai Kuang Ma 提出菱形搜索法[12]（DS）。1999 年 10 月，菱形法（DS）被 MPEG4 国际标准采纳并收入验证模型（VM）。虽然它的综合性能较其它算法优越，但它不能根据图像的内容（运动类型）作出灵活处理。即不管是什么样的运动，一律先用大模板来搜索，再使用小模板，这对小运动是一种浪费。上述这些算法通过不同的搜索模板和搜索方式，大大减少了搜索时间，但估计精度仍与 FS 有较大差距，搜索速度仍难以很好地满足实际要求。因此，有必要寻找更加高效地（快速、准确）块匹配运动估计算法。

1.3 运动估计算法研究展望

尽管块匹配运动估计算法已成为目前许多压缩标准采用的主要方法，但国际上已经开始将注意力投向可变块的运动估计算法和基于视频对象的运动估计算法研究领域。

一般的块匹配算法固定块的大小，因此存在如何选择合适的块大小问题。块太小，对于存在大范围运动的场景，常常无法估计出真实位移；而块太小，由于块内运动的不一致，可能无法得到精确的运动估值。为了解决这一问题，产生了一种可变块大小的运动估计方法——多分辨率运动估值[13]。它通过分步进行块匹配，每一步采用不同大小的块，通过一个逐渐减小的过程来实现由粗到细的运动估计，从而获得可靠性高、一致性好的运动矢量场。

MPEG4 采用现代图像编码方法，引入了视频对象平面 VOP(Video Object Plane)的概念。在这一概念中，我们根据人眼感兴趣的一些特征性如形状、运动、纹理等，将图像序列中每一帧中的场景，看成是由不同视频对象平面 VOP 所组成的，而同样对象连续的 VOP 称为视频对象 VO（Video Object），因此要求研究基于视频对象的运动估值算法，这里也包括全局运动估计（Global Motion Estimation）、前景运动物体提取、背景 Sprite 生成、处理形状（Shape）和透明（Transparency）信息等内容的研究。

MPEG4 标准的提出使得逐渐出现了许多新的基于小波理论[14]和视频对象的运动估计算法，还有其它基于遗传算法思想的运动估计方法等都各有千秋，但有一个共同的发展趋势，那就是运动估计的算法的匹配块不再局限于矩形，可以自适应改变其大小和形状，搜索路径趋于多样化，算法还要考虑硬件实现的可行性。这都是今后要进一步研究的问题。

1.4 论文的任务

论文的第一个任务是：分析研究运动估计中块匹配快速算法，并针对提高算法效率时存在的初始搜索点选择、匹配准则和搜索策略三个主要问题，分别分析讨论和比较目前常用算法的优缺点。

论文的第二个任务是：提出两种新型整像素快速算法——趋势菱形算法(TDS)和一种形状自适应快速算法（SAS）,一种新型半像素快速算法——趋势半像素算法(THS)，并应用于视频处理。研究讨论其在搜索速度、计算量、传输残差比特数、重建图像峰值信噪比等具体参数方面的情况。

论文第三个任务是：针对 VPROVE 视频分析工具原有缺陷——不支持对隔码流的解码和分析，进行 MPEG4 码流解码方面的分析研究，设计出 Video Analyzer 视频分析工具，实现对隔行码流进行解码和分析的功能，实现 VPROVE 视频分析工具的功能升级，提升 Video Analyzer 视频分析工具的应用价值。

1.5 论文的结构安排

全文共分五章，各章主要内容如下：

第一章：简单阐述了数字视频技术在通信和广播领域中的重要性，并回顾第一代、第二代数字视频编码国际标准的发展情况。

第二章：讨论分析了 MPEG4 解码框架，针对目前普遍流行的解码器只能支持 Simple Profile、Advance Simple Profile 两种档次中各种级别码流的缺陷，通过深入研究 14496-2 标准和运动补偿技术，在工程上实现了 14496-2 标准上关于 Main Profile 档次中各种级别宏块的解码，具有较高的工程价值。

第三章：讨论分析了 MPEG4 中多模式块匹配运动估计的编码框架，并提出两种新型整像素快速算法——趋势菱形算法(TDS)和一种形状自适应快速算法(SAS)，一种新型半像素快速算法——趋势半像素算法(THS)，并在各种性能参数方面与 FS 和 DS 经典算法进行对比。新算法在得到重建图像质量无明显下降的的情况下显著地减少了搜索点数，使计算量大幅度减小。

第四章：设计出视频分析工具 Video Analyzer 并简单介绍了 Video Analyzer 的各种功能，补充完善 VPROVE 视频分析工具不支持对隔行扫描的 MPEG4 解码和分析的原有缺陷，实现对隔行扫描的 MPEG4 码流进行解码和分析的功能，提升 Video Analyzer 视频分析工具的应用价值。

第五章：结论和未来工作。

第二章 MPEG4 解码框架和运动补偿技术

2.1 视频比特流语法层次

与其它视频编码标准类似，MPEG4 定义了视频比特流的生成语法和语义。在语法中必须指出如何从比特流中分解并解码生成压缩的视频信号。为了支持不同的应用，比特流必须具有灵活的语法结构，这便是具有不同层次的分层结构，每个层次有一个头信息，指示这层数据的某些特征和所采用的参数。MPEG4 比特流语法结构分层与 MPEG1/2 以及 H.261/H.263 比较如下，如表 2.1 所示。

在第一至第四的每个层次都带有开始码的头信息。这些开始码是这一层的标志信息，在视频编码数据中不允许出现这些码字。开始码之后紧跟着指示这一层编码的其它参数信息。与 MPEG1/2，类此，还定义 MPEG4 的多个档次和层次，构成了适合各种不同应用的编码子集。

表 2.1 不同标准中的语法分层结构比较

语法层次	MPEG4	MPEG1/2	H.261/H.263
第一层	Video Object Layer 视频对象层 (定义整个视频对象)	Sequence Layer 视频序列层 (定义整个视频序列)	Sequence Layer 视频序列层 (定义整个视频序列)
第二层	Group VO Plane Layer 视频对象平面组层 (支持视频流的随机存取)	Group of Picture 图像组层 (支持视频流的随机存取)	
第三层	VO Plane Layer 视频对象平面层 (主要编码单元)	Picture Layer 图像层 (组要编码单元)	Picture Layer 图像层 (组要编码单元)
第四层	Video Packer 视频打包层 (用于重同步和误码恢复)	Slice Layer 图片层 (用于重同步、刷新和误码恢复)	Group of Block 块组层 (用于重同步、刷新和误码恢复)

第五层	MacroBlock Layer 宏块层 (运动补偿和形状编码单元)	MacroBlock Layer 宏块层 (运动补偿单元)	MacroBlock Layer 宏块层 (运动补偿单元)
第六层	Block 块层 (变换和补偿、填充单元)	Block 块层 (变换和补偿、填充单元)	Block 块层 (变换和补偿、填充单元)

2.2 视频解码过程

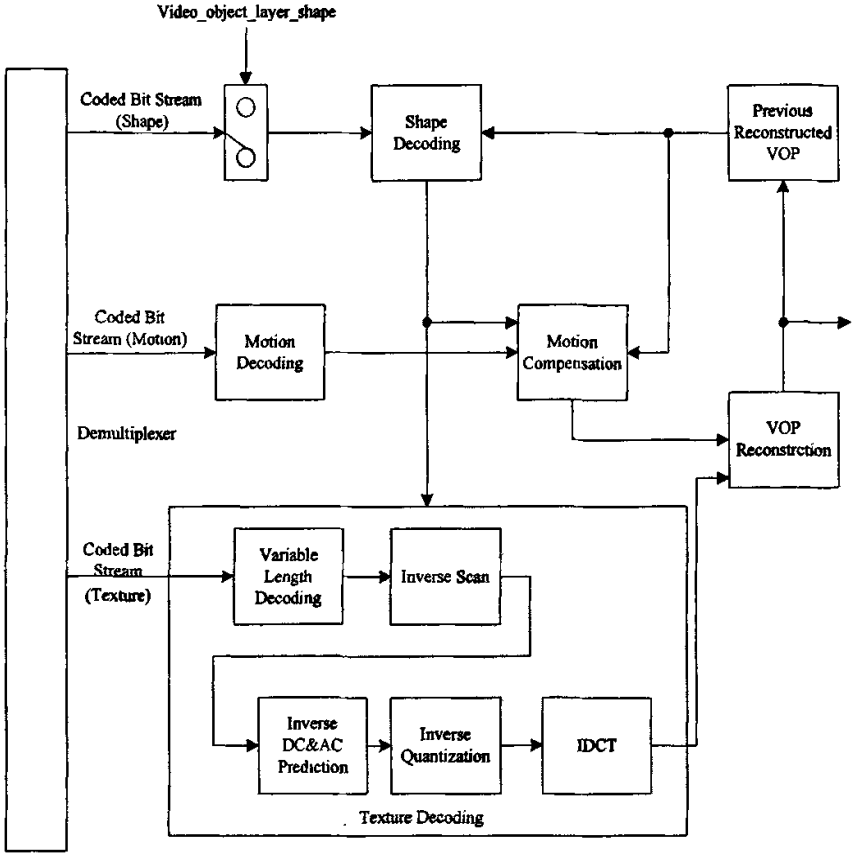

图 2.1 简化的视频解码过程

除开逆离散余弦（IDCT），所有的解码器应该产生相同的结果,以下的解码过程都是按着如图 2.1 视频解码过程依次进行解码。

2.3 帧内视频对象平面的解码

因为是 IVOP 为帧内编码，依次对当前帧每个宏块进行以下解码操作。

解码帧内视频对象平面 IVOP 中当前宏块的纹理头部信息。先计算出宏块中非透明亮度块的数目，为计算亮度块编码模式提供依据。再哈夫曼解码得到索引值即宏块类型色差编码块模式 MCBPC。如果 MCBPC 大于 3，那么宏块类型为 INTRAQ；否则就为 INTRA。而 MCBPC 余 4 操作得到色差块编码模式 CBPC。接着解码一比特值即 AC 预测标志符，即当这个标志符为 1 对帧内宏块的第一行或第一列进行水平或垂直预测。利用非透明块个数进行相应的哈夫曼解码得到亮度块编码模式 CBPY。再利用 CBPY、CBPC、宏块模式、非透明块个数进行亮度和色差编码块模式设置为下面的解码提供参数，依次给 4 个亮度块和 2 个色差块设置编码块模式。如果宏块类型为 INTRAQ，则解出 2 比特得到一索引值，通过它查表得到差分量化步长即 StepDelta 值，它表示相邻宏块之间的量化步长是一直变化的而不是一个常量。用 StepDelta 值加上量化步长箝位后就是当前宏块的量化步长。

再解码帧内宏块纹理。先对 Qp 大小进行判断后可对亮度和色差 DC 缩放比例赋值。然后对当前宏块的 4 个亮度块和 2 个色差块进行处理。

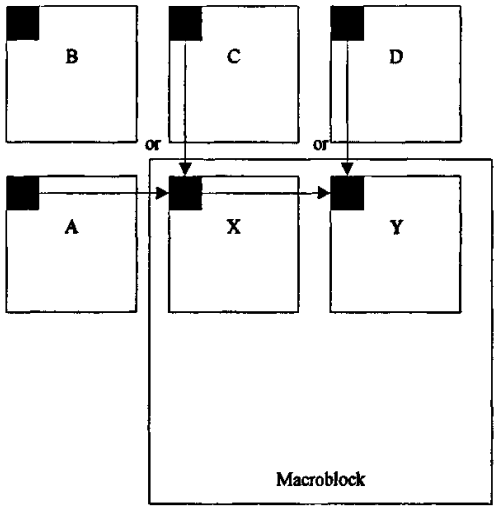

图 2.2 先前相邻块用于当前块 DC 预测

然后确定帧内直流和交流预测方向，如图 2.2 所示。判断当前块从哪个方向进行帧内预测。先计算出垂直方向、水平方向直流预测值和对角方向直流预测值。

如果左上角块直流预测值为空，则左上角预测值赋给一个缺省值。如果顶部块直流预测值为空则用缺省值、不为空则用顶部块直流预测值，减左上角块直流

预测值得到的结果赋给水平格点；如果左边块直流预测值为空则将缺省值、不为空则用左边块直流预测值，减左上角块直流预测值得到的结果赋给垂直格点。

如果垂直格点的绝对值小于水平格点的绝对值，则当前块的预测方向赋为垂直方向，如果顶部块直流预测值不为空，则将顶部块直流预测值赋给预测块直流预测值，顶部的预测量化步长赋给预测量化步长；如果垂直格点的绝对值大于或等于水平格点的绝对值，则当前块的预测方向赋为水平方向，如果左边块直流预测值不为空，则将左边块存储器赋给预测块直流预测值，左边的预测量化步长赋给预测量化步长。

再进行解码帧内块纹理。先解码帧内当前块的直流残差系数，然后返回这个帧内 DC 差分。再判断当前块编码块模式中对应块是否为 0：如果为 0 则这个 8×8 块中的交流系数全赋给 0 值；如果不为 0，则选取扫描方式，然后解码这个块剩下的 63 个交流系数。解码帧内块系数，先判断 LAST 标志是否为 1，1 则表示这个块后面的所有系数全为 0，就将所剩的系数全部赋给 0 值；不为 0，进行哈夫曼解码得到一个索引值，如果这个索引值不为 102，则通过这个索引值查帧内可变长索引表即用 LEVEL 的掩码值与上以索引值为索引的帧内可变长哈希表中的值后就得到了 LEVEL 的值。用 RUN 的掩码值与上以索引值为索引的帧内可变长哈希表中的值，之后要右移 5 位，就得到了 RUN 的值。用 LAST 的掩码值与上以索引值为索引的帧内可变长哈希表中的值，之后要右移 10 位，就得到了 LAST。接着解码 1 比特即 LEVEL 的符号位，这样就求出了 LEVEL、RUN、LAST。如果这个索引值为 102 值，进入解 Escape 码。直至 LAST 为 1，则这个系数矩阵后面的所有系数赋给 0 值，这样就完成一个帧内块残差系数的解码。

接着进行逆交直流系数预测。如图 2.3 所示，当直流预测值为空时，将缺省值 1024 加上，DC 缩放比例值右移一位，再除以 DC 缩放比例值，取整后的值加上解码出来的直流残差值即为当前块真真的直流系数；否则将缺省值换成直流预测值即可再进行直流系数箝位，接下来处理第一行或第一列的系数。如果当前块的预测方向为水平方向时，可以得到预测交流系数中第 8、16、24、32、40、48、56 个系数的值。当预测块为空则预测交流系数中相应的值为 0，否则再判断当前块的量化步长是否等于预测块的量化步长，如果相等则可以取到预测交流系数相应的值，否则将预测块中相应的交流预测值乘以预测量化步长后加上当前量化步长右移一位，再除以当前量化步长，取整后的值作为预测交流系数中相应的值。预测值加上残差值后得到的交流系数再箝位；如果当前块的预测方向为垂直方向时，可以得到预测交流系数中的第一行，过程和水平方向预测相似。

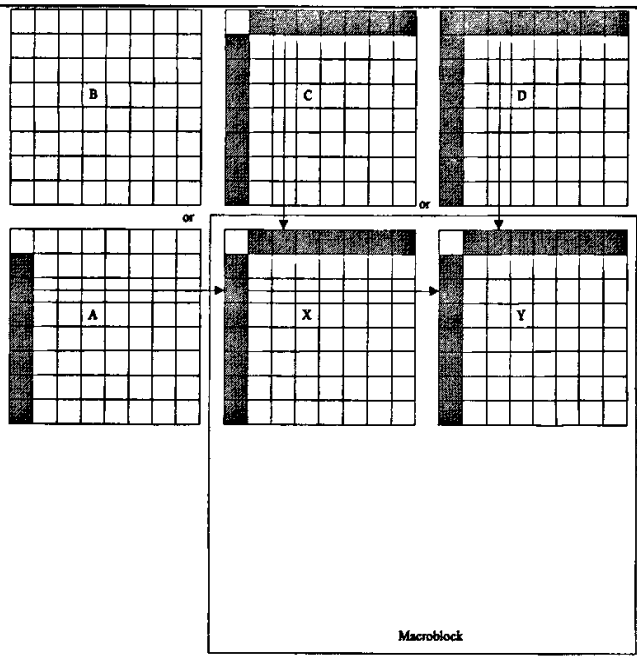

图 2.3 先前相邻块用于当前块 AC 预测

接着进行帧内 DCT 变换系数的饱和处理。再进行帧内 DC 值的逆量化。将变换系数的第一个元素值乘以 DC 的缩放比例值，再进行箝位即直流部分。再选择相应的逆量化器进行 DCT 系数的量化。接着将这个 8×8 的系数矩阵进行 IDCT 变换，得到一个 IDCT 变换后的矩阵。当解码完 6 个块的纹理时，解码完成即得到当前宏块解码后的图像数据。

2.4 帧间视频对象平面 PVOP 的解码

解码 PVOP 中当前宏块纹理头部信息。计算出宏块内完全不透明亮度块的数目，为后面计算亮度块编码模式提供依据。再解码一位即 SKIP 标志符，如果为 1 即当前宏块没有被编码，则当前宏块类型为 INTER；如果为 0 即当前宏块被编码过，哈夫曼解码出 MCBPC 的索引值，索引值除以 4 取整即宏块类型，如果为 0 那么宏块类型就为 INTER 且 4MV 标志为 0；如果为 1 那么为 INTERQ 且 4MV 标志为 0；如果为 2 那么为 INTER 且 4MV 标志为 1；如果为 3 那么为 INTRA；如果为 4 那么为 INTRAQ。MCBPC 余 4 得到色差块编码类型。如果宏块类型为 INTRA 或 INTRAQ 即采用帧内编码方式，则解码一位即 AC 预测标志符。接着哈夫曼解码出帧内的 CBPY；其它则根据全不透明块数数目不同分 4 中情况，分别通过哈夫

曼解码获取相应 CBPY 的索引值，查表得到 CBPY。再利用 CBPY、CBPC、非透明块亮度块数依次给 6 个块设置是否被编码的标志。如果宏块类型为 INTERQ 或 INTRAQ,则解出 2 位即一个索引值，即 4 个不同 StepDelta 值，用 StepDelta 加上量化步长就是当前宏块的量化步长。如果当前 VOP 是隔行的且 SKIP 为 0、且当前宏块类型为 INTERQ 或 INTER、且不为 4MV 类型，那么解码一比特作为当前宏块帧间场预测的标志。如果此标志符为 1，那么依次解码一位作为当前宏块前向顶场参考底场的标志，接着解码一位作为当前宏块前向底场参考底场的标志。

PVOP 运动矢量解码过程如图 2.4 所示：

第一种情况 SKIP 为 1 或宏块类型为 INTRA 或 INTRAQ，则无运动矢量。

第二种情况 4MV 标志为真，则依次解码这 4 个 8×8 子块的运动矢量。如果当前宏块位于 VOP 的边界，则开始找 MV 的一般性的预测值。如果当前宏块不位于 VOP 的边界，则开始找内部宏块的 MV 预测值。而运动矢量预测值为三个候选块运动矢量水平和垂直分量的中值。再解码每个块的运动矢量差分。将差分运动矢量加上预测运动矢量值就得到当前亮度块的运动矢量，再进行箝位。

第三种情况有一个 MV，则先计算预测运动矢量即三个候选宏块运动矢量水平和垂直分量的中值。再解运动矢量差分，将差分运动矢量加上预测运动矢量值就得到整个宏块的运动矢量，再进行箝位。

第四种情况如果当前 VOP 是隔行且当前宏块要用前向场运动矢量做补偿，先计算场预测运动矢量即三个候选块运动矢量水平和垂直分量的中值。再解码运动矢量差分，只是将运动矢量差分的垂直分量乘以 2，再将差分运动矢量加上预测运动矢量值就得到整个宏块的前向顶场运动矢量 FTMV 再将箝位。如果 FTMV 参考了底场，将 FTMV 放到宏块的运动矢量序列中 FTBMV 的位置上；如果 FTMV 参考了顶场，将 FTMV 放到 FTTMV 的位置上。再进行一次运动矢量差分解码，也将差分运动矢量的垂直分量乘以 2，再将差分运动矢量加上预测运动矢量值就得到整个宏块前向底场的运动矢量 FBMV 再进行箝位。如果当前宏块 FBMV 参考了底场，将 FBMV 放到 FBBMV 的位置上；如果当前宏块前向底场 FBMV 参考了顶场，将 FBMV 放到 FBTMV 的位置上。

如果是帧内编码则进行帧内宏块纹理解码和 IVOP 中解码方法一样。

接着进行帧间宏块纹理解码即依次对 4 个亮度块和 2 个色差块进行解码。如果当前块没有被编码那直接给相应块的残差值赋 0；否则进行帧间块的纹理解码，进行帧间可变长码表索引查表，这样可以确定 RUN、LEVEL、LASTRUN 这三个标志符。直至 LASTRUN 为 1 则表示这个系数矩阵后面的所有系数赋给 0 值。

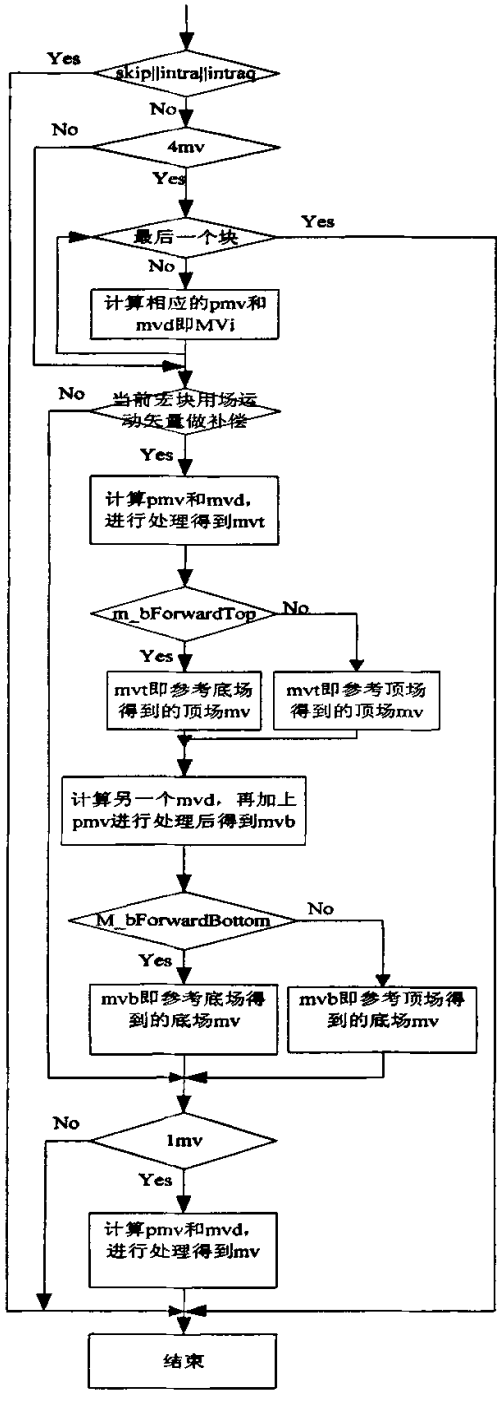

图 2.4 PVOP 运动矢量解码过程

接着进行逆量化，过程和 IVOP 相似。再进行 IDCT 变换，这里所得到的系数

才是运动估计后的残差值。当解码完六个块时，当前宏块也解码结束。

如SKIP就不进行运动补偿，直接从参考帧中相应位置的宏块拷贝给当前宏块。

如不是 SKIP 类型宏块那么进行运动补偿，首先将 1/2 像素为单位位置参考点加上运动矢量的大小限制在一个范围内可以找到当前块在参考帧中的最佳匹配位置，大小可以是 16×16 大小的宏块，也可以是 8×8 大小的亮度块。

对于 1MV 的 1/2 像素的运动补偿。

第一种情况最佳匹配点在参考帧中为整像素位置。将参考位置处中的 16 个像素点的值复制到预测宏块中，而参考宏块指针向后偏移一帧的宽度，而预测块的指针只向后偏移 16 个像素。如上所述直至最后一行。

第二种情况最佳匹配点在参考帧中垂直位置为整像素而水平位置为半像素。将参考块第一行第一个元素加上第二个元素值再凑整后往右移 1 位即预测块第一行第一个元素，接着将参考块第一行第二元素加上第三个元素值再凑整后再往右移 1 位即预测块第一行第二个元素。依此类推直到最后一行。

第三种情况最佳匹配点在参考帧中垂直位置为半像素而水平位置为整像素。将参考块第一行第一个元素加上第二行第一个元素值再凑整后往右移 1 位即预测块第一行第一个元素，接着将参考块第一行第二元素加上第二行第二个元素值再凑整后往右移 1 位即预测块第一行第二个元素。依此类推直到最后一行。

第四种情况最佳匹配点在参考帧中为半像素。从第一行开始，将参考块第一行第一个元素加上第二个元素再加参考块第二行第一个元素加上第二个元素再凑整后往右移 2 位即预测块第一行第一个元素，接着将参考块第一行第二个元素加上第三个元素再加参考块第二行第二个元素加上第三个元素再凑整后往右移 2 位即预测块第一行第二个元素。依此类推直到直至最后一行。

对于 4MV 的 1/2 像素的运动补偿。过程如 1MV 的 1/2 像素的运动补偿，只是要进行位置的调整即相应的块加上各自的运动矢量指向参考帧中最佳匹配点。

对于 1MV 的 1/4 像素的运动补偿。首先将 1/4 像素为单位的参考点加上运动矢量限制在一个范围内即参考帧中最佳匹配位置。接着进行亮度块 1/4 内插，先将最佳匹配位置用整数和小数部分表示，并根据 1/4 精度点的位置往向上或向左或向左上角偏移最多 1 个整像素点即包含真真最佳匹配点，从参考帧中拷贝出一个 17×17 的矩阵当作参考块，以下的操作都是针对这个参考块。先进行水平镜像扩展构造成一个 17×23 的水平扩展参考块。如果水平位置上的小数部分为 0 时即最佳匹配点是参考块左上角的水平位置，直接将水平扩展参考块中从第 4 行第 4 列开始的一个 17×16 矩阵赋给从第四行第一列开始的一个 23×23 水平临时矩阵；如

果为 1 时即发生 1/4 像素位移，那么将经水平扩展参考块中每行每 8 个像素值进行 8 抽头 FIR，得到一个 17×16 的矩阵，依次再将这个矩阵与水平扩展参考块中从第 4 行第 4 列开始相对应位置的像素值的和加 1 再经过凑整除以 2，赋给从第四行第一列开始的一个 23×23 水平临时矩阵；如果为 2 时即发生 2/4 像素位移，那么将经水平扩展参考块中每行每 8 个像素值进行 8 抽头 FIR，得到一个 17×16 的矩阵，将它赋给从第四行第一列开始的一个 23×23 水平临时矩阵；如果为 3 时即发生 3/4 像素位移，将经水平扩展参考块中每行每 8 个像素值进行 8 抽头 FIR 再凑整箝位，得到一个 17×16 的矩阵，再将这个矩阵与经过水平扩展矩阵的第四行第五列开始相对应位置的像素值的和加 1 再凑整除以 2，得到一个 17×16 的矩阵，将它赋给从第四行第一列开始的一个 23×23 水平临时矩阵。再将这个 23×23 水平临时矩阵从 0 至 15 列分别向上和向下扩展 3 行，而 16 至 23 列不用，得到一个 23×16 垂直扩展矩阵，如果垂直位置上的小数部分为 0 时即在垂直方向上最佳匹配点与参考块左上角的垂直位置匹配点重合，将这个 23×16 垂直扩展矩阵，从 4 到 19 行 1 到 16 列的矩阵当作 1/4 像素预测块。如果为 1 即在垂直方向上发生了 1/4 像素位移，将这个 23×16 垂直扩展矩阵，做垂直方向 8 抽头 FIR 再凑整箝位，得到一个 16×16 的临时矩阵，再将这个临时矩阵与 23×16 垂直扩展矩阵从 4 行 1 列到 19 行 16 列相应的元素凑整除以 2，得到的这个矩阵就是 1/4 像素预测块；如果为 2 即在垂直方向上发生了 2/4 像素位移，将这个 23×16 垂直扩展矩阵做垂直方向 8 抽头 FIR 再凑整箝位，得到一个 16×16 的临时矩阵即 1/4 像素预测块；如果为 3 时即发生了 3/4 像素位移，将这个 23×16 垂直扩展矩阵，做垂直方向 8 抽头 FIR 再凑整箝位，得到一个 16×16 的临时矩阵，再将这个临时矩阵与 23×16 垂直扩展矩阵从 5 行 1 列到 20 行 16 列相应的元素相加并加 1 凑整除以 2，得到的这个矩阵就是 1/4 像素预测块。

对于 4MV 的 1/4 像素的运动补偿。过程如 1MV 的 1/4 像素的运动补偿一样，只是进行 8×8 块的运动补偿，并用相应子块的运动矢量，通过运动矢量和坐标值可以得到参考帧中最佳匹配位置，得到的预测值放在相应的预测亮度块中。

如果当前宏块要用场运动矢量做运动补偿。那么用 FTT 即当前顶场用前向顶场作参考的运动矢量。如果前向顶场参考底场标志为真，那么当前场的运动矢量用 FTB，那么参考帧中左上角的位置还要向下移动一行以便指向底场；如果当前前向顶场参考底场标志为假，那么还是指向 FTT。再判断是否为 1/4 或 1/2 精度的场运动补偿。如果是 1/4：场运动补偿时水平方向的长度是 16 个像素点，垂直方向的长度是 8 个像素点。接着进行亮度块的双线性内插，由于是场编码的所以每

处理完一行后，要往下跳 2 行的间隔。被当作参考块的是由各种参数定位后的一个 9×17 的矩阵，和上面帧所述的双线性内插基本上一样只是在垂直方向上的位移是 2 倍大小，最后得到一 8×16 的预测块，再将这个预测块赋给一个 16×16 的矩阵，每赋值完一行这个矩阵往后偏移一行，用来存放底场的数据。处理完顶场后，预测块的指针往后移动 16 个像素值即指向底场的像素位置，再判断前向底场参考底场标志是否为真，如果为真则用 FBB，而参考帧是底场；如果不是则用 FBT。1/2 场运动补偿：在垂直方向上的位移都是 2 倍大小，先后判断前向顶场是否参考底场、前向底场是否参考底场，由此确定用参考场的底场还是顶场。再根据运动矢量和参考位置确定的值在参考帧中位置的不同有四种处理，过程和 1MV 帧的 1/2 像素亮度的运动补偿一样，只是处理完一行后，预测块的指针位置往后移动 32 个像素点即跳到同一场的下一行，而参考帧的指针往后移动 2 行帧的宽度。

从亮度块的运动矢量计算 U、V 块的运动矢量。如果 1/4 取样和 4MV 标志都为真，那么将四个亮度块运动矢量的水平和垂直分量分别累加在一起，将得到的两个分量分别进行绝对值整数除以 16 再乘以 2、加上绝对值对 16 求余后查表 grgiMVRound16 的值，再乘以各自的符号即 U、V 的运动矢量。如果是 1MV 帧编码的宏块时，将水平和垂直两个分量分别进行绝对值整数除以 4 再乘以 2、加上绝对值对 4 求余后查表 grgiMVRound4 的值，再乘以各自的符号即 U、V 的运动矢量。

如果作色差 U、V 帧的运动补偿。以 1/2 像素为单位，参考 U、V 平面的左上角位置再偏移运动矢量的水平、垂直分量分别加上当前块坐标的水平、垂直分量后都要往右移动一位后即当前块在参考帧中 U、V 最佳匹配位置，再根据在参考帧中位置的不同有四种不同的处理，过程和 1MV 帧 1/2 像素亮度的运动补偿一样。

如果作色差 U、V 场的运动补偿。以 1/2 像素为单位，参考 U、V 平面的左上角位置再偏移坐标的水平分量加上色差矢量的水平分量后往右移动一位、而垂直分量为坐标的垂直分量除以 2 加上、色差矢量的垂直分量后往右移动一位，接着还要加上顶场或底场的偏移量，于是就得到了的色差最佳匹配参考点的位置，再根据 U、V 指针位置的不同进行运动补偿，过程和色差 U、V 帧的运动补偿一样，只是都要用同一场的元素并且处理完一场后参考 U 和 V 指针向后偏移 2 倍帧宽的一半，而预测 U 和 V 指针只向后偏移 16 个像素的大小，空出来的 8 个点用来放另外一场的像素值。依此类推接下来的 7 行。

将处理得到的预测数组和残差数组对应的像素值相加并箝位后赋给当前宏块中相应位置的亮度和色差像素值。

2.5 BVOP 的解码

从当前帧的第一个宏块开始解码，如果 SKIP 的标志为真，那么不进行解码直接将参考帧中的 YUV 数据分别拷贝到当前宏块中。

如果 SKIP 的标志为假，就对 BVOP 中宏块头部纹理信息进行解码。再利用 CBPY、CBPC、宏块编码模式、非透明块数进行亮度块和色差块编码模式的设置。

最后判断隔行标志符为真则解码一位赋给场预测的标志。如果隔行扫描标志符为 1 且不为 DIRECT，则解码一位为运动补偿是否用场运动矢量标志符。如这个标志符为 1 且不为 BACKWARD，则解出两位值依次赋给前向顶场参考底场的标志、前向底场参考底场的标志；如果为 1 且宏块类型不为 FORWARD，则从码流中解出两位值依次赋给中后向顶场参考底场的标志、后向底场参考底场的标志。

如图 2.5 BVOP 运动矢量的解码过程如下：

第一种情况 SKIP 标志为 1，那么不需要运动矢量的解码。

第二种情况宏块类型为 FORWARD 或 INTERPOLATE，那么解码差分运动矢量 MVD。如果场运动矢量用于运动补偿标志符为真，直接将 MVD 加上前向顶场预测运动矢量其中垂直分量要整数除以 2，箝位后再将垂直分量乘以 2，即当前宏块的前向顶场运动矢量 FTMV。再进行解码 MVD，将 MVD 分别加上前向底场预测运动矢量过程和求 FTMV 一样，得到当前宏块的前向底场运动矢量 FBMV；否则将 MVD 加上前向预测运动矢量赋给当前宏块的前向运动矢量 FMV。

第三种情况宏块类型为 BACKWARD 或 INTERPOLATE，则解码 MVD。如场运动补偿标志符为 1，则将 MVD 加上预测后向顶场运动矢量其中垂直预测分量要整数除以 2，并将垂直分量乘以 2 后，赋给 BTMV。再进行解码 MVD 后，将 MVD 加上后向底场预测运动矢量，过程和求 BTMV 一样赋给 BBMV；否则直接将 MVD 加上后向预测矢量赋给 BMV 即后向运动矢量。

第四种情况宏块类型为 DIRECT。如果隔行扫描标志为真，又分为两种情况。

第一种情况 SKIP 和 1/4 标志都为假，解码差分运动矢量得到直接 DeltaMV。如果 SKIP 为假且 1/4 精度标志为真，进行哈夫曼解码一个索引符号值，而 deScaleMV 直接得到 DeltaMV。第二种情况 SKIP 为真，直接给 MVD 的赋 0。

计算 DIREC 模式的前向运动矢量，分为两种情况。

第一种情况参考运动矢量为空，则直接将 MVD 赋给当前块的运动矢量，即四个前向直接运动矢量都等于 MVD。

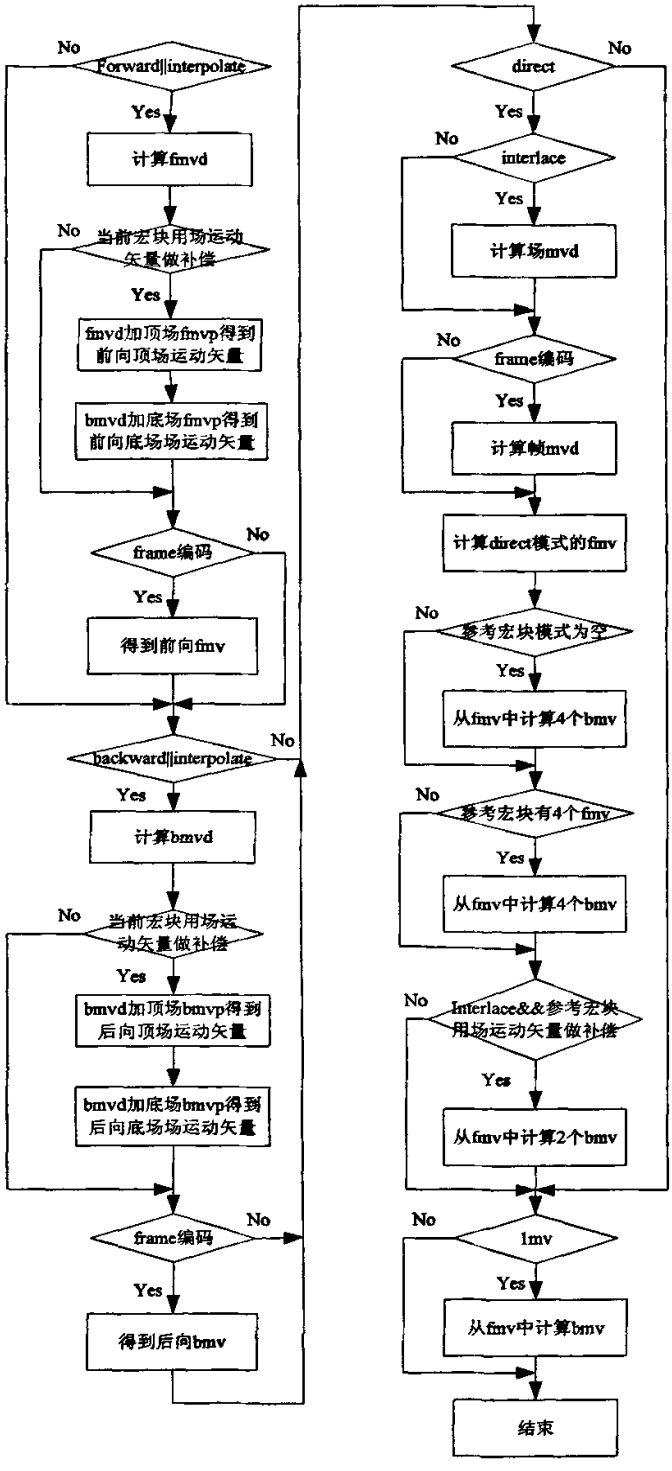

图 2.5 BVOP 运动矢量的解码过程

第二种情况参考运动矢量不为空且前向 4MV 的标志为真，TRB 等于当前时间减去最近过去前向参考帧的时间 ，TRD 等于最近将来后向参考帧的时间减去最近过去前向参考帧的时间。然后在 4 个块中依次进行如下操作：将参考运动矢量的各分量乘上 TRB/TRD 得到的值加上 MVD 赋给 MV，即 FMVi。

如果参考运动矢量不为空且、当前帧是隔行扫描且参考宏块模式中用场运动矢量作运动补偿的标志为真。则 TRB、TRD 作一番处理后，在参考运动矢量序列中取参考宏块顶场参考了底场的运动矢量 TBMV。

如果参考宏块模式中前向顶场参考底场的标志为真，再进行判断当前宏块顶场第一标志是否为真，如果为真，则 TRB--、TRD--；否则 TRB++、TRD ++,因为参考宏块模式中前向顶场参考底场，所以参考运动矢量是 TB，将参考运动矢量分量乘以 TRB/TRD 后加上 MVD 即当前直接模式的前向顶场运动矢量 FTMV。

如果参考宏块模式中前向顶场参考底场的标志为假，将直接将参考运动矢量乘以 TRB/TRD 后加上 MVD 后即当前直接模式的前向顶场运动矢量 FTMV。

以上是解码前向顶场运动矢量，接下来是前向底场运动矢量。

如果参考宏块中前向底场参考底场的标志为真，将直接将参考运动矢量乘以 TRB/TRD 后加上 MVD 后即当前直接模式的前向底场运动矢量 FBMV。

如果参考宏块中前向底场参考底场的标志为假，再进行判断当前 VOP 模式中顶场第一标志是否为真，如果是 TRB--、TRD--；否则 TRB++、TRD++，参考运动矢量是 BT，将参考运动矢量分量乘以 TRB/TRD 后加上 MVD 即 FBMV。

其它情况下即一个运动矢量用同样的方法求 TRB、TRD，从而求得 FMV。

接下来处理直接模式下的后向运动矢量，也分为两种情况。

第一种情况宏块模式参考为空。利用参考运动矢量和当前的 MVD，从 FMV 中求 BMV。如果 MVD 的水平分量等于 0，那么将-TRB/TRD 乘以参考运动矢量的水平分量赋给后向运动矢量的水平分量；如果 MVD 的水平分量不等于 0，那么将当前 FMV 减去参考运动矢量水平分量赋给 BMV 的水平分量 。同理可得垂直分量 。再依次完成从四个亮度块的 FMV 中得到 BMV。

如果参考宏块模式不为空，那么将参考宏块模式中四个前向 MV 标志为真，即帧编码的 4 个直接后向运动矢量，则依次将四个亮度块作一次从 FMV 求 BMV 的运算；如果宏块模式参考不为空且、当前隔行标志为真且参考宏块模式中用场运动矢量作补偿的标志为真，那么作二次从 FMV 求 BMV 的计算，分别得到 BTMV 和 BBMV。其它情况即只有一个运动矢量，则直接作一次从 FMV 求 BMV 的计算。

对于解码 BVOP 中的纹理，过程和 PVOP 种的解码过程一样。接着恢复原始

数据——预测数据加上残差数据。

第一种情况隔行扫描标志为真。如果宏块类型是 FORWARD，那么只做前向运动补偿：有当前宏块前向顶场参考底场和前向底场参考底场标志即 topRef and botRef。如果用场运动矢量作补偿的标志为 1 时，那么顶场和底场的运动矢量分别为*pmvTop=pmv+1+topRef,*pmvBot=pmv+3+botRef，在这里其中 pmv 是宏块运动矢量，而 pmvTop 是顶场的运动矢量，依次先作顶场运动补偿，再作底场运动补偿。得到真正的半像素运动矢量，接着对色差 UV 分量做场的运动补偿。

再作底场运动补偿：可以对亮度块的底场进行 1/4 或 1/2 像素运动补偿。接着再对色差块的底场进行 UV 场的运动补偿。

如果宏块类型是 BACKWARD，分别进行后向参考帧的运动补偿、将预测值加上残差值赋给当前宏块，如上面 FORWARD 过程所述。

Direct 模式的运动补偿：通过参考宏块模式加上宏块行列偏移量就可以得到当前宏块的参考宏块模式。先计算当前宏块顶场和底场各自运动矢量的：前向顶场运动矢量为参考宏块的顶场运动矢量作一次线性运算再加上当前直接模式的 DeltaMV；前向底场运动矢量为参考宏块的底场运动矢量作一次线性运算再加上当前直接模式的 DeltaMV；后向顶场运动矢量为，如果当前直接模式的 DeltaMV 不为 0，那么将前向顶场的运动矢量减去参考宏块的顶场运动矢量；否则将参考宏块顶场运动矢量作一次线性运算；后向底场运动矢量为，如果当前直接模式的 DeltaMV 水平分量不为 0，那么将前向底场的运动矢量减去参考宏块的底场运动矢量；否则将参考宏块底场运动矢量的水平分量作一次线性运算；接着进行前向顶场 1/4 像素或 1/2 像素的运动补偿。接着进行色差块的前向顶场运动补偿。再进行后向顶场运动补偿。接着进行色差块的后向顶场运动补偿。接着进行前向底场运动补偿。再进行色差块的前向底场运动补偿。接着进行后向底场运动补偿。再进行色差块的后向底场运动补偿。

根据 uiDivisor 的值选取不同的 grigiMvRoundx。将前向色差运动矢量取绝对值按 uiDivisor 取余后得到值为索引在 grigiMvRoundx 查表取值，再加上前向色差运动矢量取绝对值整数除以 uiDivisor 后乘以 2，再乘以前向色差运动矢量的符号值，将所得值即前向色差运动矢量。同理可得后向色差运动矢量。

如果只有一个参考运动矢量。那么过程和参考宏块模式中前向 4MV 标志为真一样，只是对整个宏块进行运动补偿而不是针对每个亮度块。

如果没有参考宏块。将直接方式的 MVD 赋给 FMV；如果 FMV 的水平分量不为 0，直接将 FMV 的水平分量赋给 BMV 的水平分量，如果 FMV 的水平分量为 0，

直接给 BMV 的水平分量置 0；垂直分量类似。然后以块为单位做当前宏块前向和后向最近参考帧的 1/4 或 1/2 像素精度的运动补偿。而色差块的运动矢量的求法和只有一个参考运动矢量中的一样。平均两个预测块加上一个残差块赋给当前宏块。

如果宏块类型是 INTERPOLATE，先后对前向和后向参考宏块做运动补偿，进行平均前向预测和后向预测后加上当前宏块的残差值赋给当前宏块。

第二种情况隔行扫描标志为假。

如果宏块类型为 DIRECT 或 INTERPOLATE：如果是 INTERPOLATE 的一个前向 MV，再进行 1/4 或 1/2 的运动补偿。如果是 DIRECT 的前向四个 MV，则对四个亮度块依次进行 1/4 或 1/2 的运动补偿。接下来进行前向的 U、V 色差运动矢量的计算，再进行色差 U、V 块的运动补偿处理。与 PVOP 中色差块的运动补偿。

如果是后向 INTERPOLATE 的一个后向 MV，则进行 1/4 或 1/2 的运动补偿；如果是 DIRECT 的后向 4MV，则对四个亮度块依次进行 1/4 或 1/2 的运动补偿。接下来进行后向的 U、V 色差运动矢量的计算，再进行色差 U、V 块的运动补偿处理。取前向和后向预测值的平均加上残差值赋给当前像素点的亮度值。

如果宏块类型为 FORWARD 或 BACKWARD，就进行前向或后向的 1/4 或 1/2 的运动补偿。接下来进行从亮度运动矢量计算和查表得到色差块的运动矢量，再进行色差 U、V 块的运动补偿处理。取前向或后向预测值加上残差值赋给当前像素点的亮度值。

2.6 本章小结

本章讨论分析了 MPEG4 解码框架，针对目前普遍流行的解码器只能支持 Simple Profile、Advance Simple Profile 两种档次中各种级别码流的缺陷，通过深入研究 14496-2 标准和运动补偿技术，在工程上实现了 14496-2 标准上 Main Profile 档次中各种级别的前向和双向隔行的视频对象平面中类型为 INTER、INTERPOLATE、FORWARD、BACKWARD、DIRECT 宏块的解码，具有较大的工程价值。

第三章 MPEG4 编码框架与运动估计的快速算法

3.1 运动估计的基本原理

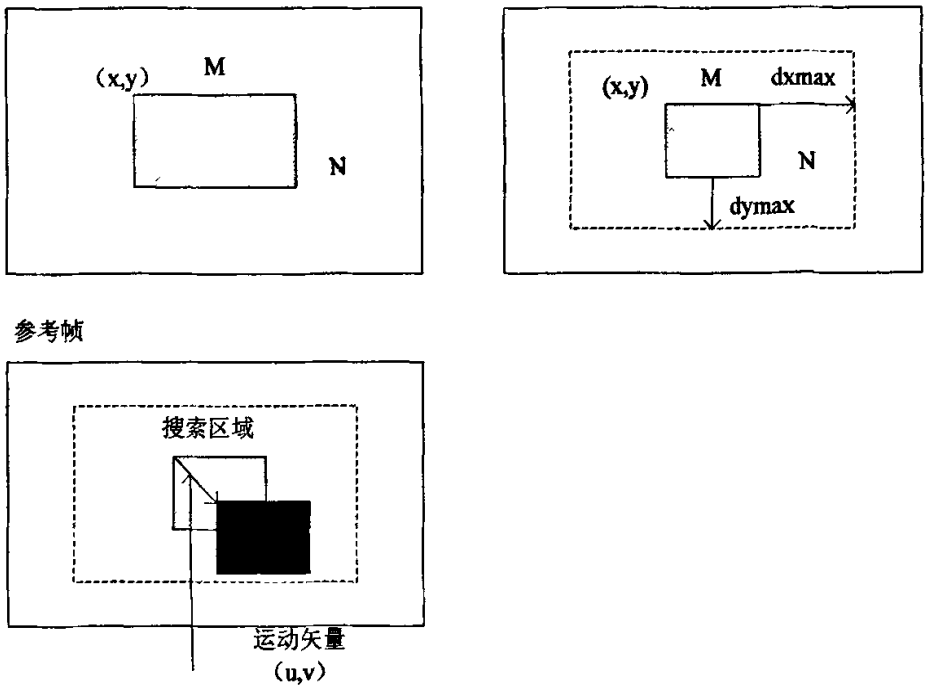

图 3.1 运动估计的基本原理

如图 3.1 所示，运动估计的基本思想是将图像序列的每一帧分成许多互不重叠的宏块[2]，并认为宏块内所有像素的位移量都相同，然后对于当前帧中的每一块到前一帧或后一帧某一给定搜索范围内根据一定的匹配准则找出与当前块最相似的块，即匹配块，由匹配块与当前块的相对位置计算出位移即为当前块的运动矢量。

宏块大小为 M×N，一般取 16×16。搜索范围一般由最大偏移矢量来决定，设可能的最大偏移矢量为(dx_{max}, dy_{max})，则搜索范围可以表达为：

$$(M + 2 \times dx_{max}) \times (N + 2 \times dy_{max}) \tag{3-1}$$

常用的匹配准则有最小绝对差（MAD）、最小均方误差（MSE）和归一化互相关函数（NCCF）。

3.2 MPEG4 中多模式块匹配运动估计的编码框架

3.2.1 前向运动估计编码框架

对 PVOP 的运动估计：对于当前帧中的各个宏块都做如下处理。

先对当前宏块与相对应参考宏块做绝对差值和，得到一个差值 SAD0。

接着进行 16×16 块的匹配：在最大搜索范围内对每一个像素点进行 16×16 块的匹配。值得注意的细节，每完成一行绝对差值和就跟 SAD0 比较，如果大于等于 SAD0 值，那么就停止比较，对下一个像素进行 16×16 块的匹配；如果小于 SAD0 值，那么就进行第二行的绝对差值和，再和 SAD0 比较，直到完成最后一行，如果还是小于 SAD0 值，那么将所到的 SAD 值作为 SAD0，而将当前的位置作为最佳匹配点，再进行下一个像素点的 SAD 值比较。最后得到最佳整像素匹配点。

如果不是 1/4 像素取样，先找到可能的 8 个半像素点。往左边或上边或左上角偏移半个像素，逐一以最佳匹配整像素周围 8 个半像素点为顶点的 16×16 宏块，与当前宏块进行 SAD 计算，过程与上述过程一样，但是每处理完一行后扩展参考指针要往后偏移 2 个扩展亮度块的宽度。得到的最小 SAD 点即最佳半像素匹配点。

如果是 1/4 像素取样，先进行帧间亮度块内插，可以得到一个 16×16 由参考帧插值形成的矩阵，然后将这个矩阵与当前编码宏块进行 SAD，依次比较 8 个半像素点找到最小的 SAD 值即最佳匹配半像素点，因为现在要得到 1/4 像素的运动矢量，所以整像素乘 4 加上半像素乘 2 即以 1/4 精度表示。再以最佳半像素点为中心在它周围 8 个 1/4 精度点逐一进行亮度块双线性插值得到一个内插块，再对这个内插块和当前编码块进行 SAD，当 SAD 值最小的点即 1/4 像素精度最佳匹配点。再进行偏移量的计算：对整个编码宏块 256 个像素值求平均值，依次对每个像素值减去这个平均值求和即对平均值偏移量之和。

如果是场编码，那么先进行前向顶场参考顶场 16×8 的匹配：将编码块和参考宏块指向相应的场，进行 16×8 矩阵的 SAD 计算，每做完一行，参考指针往下偏移 2 个亮度帧宽度，得到一个没有位移的 SAD0，然后在最大搜索范围内进行螺旋搜索法，只有垂直方向的坐标值和垂直方向的位移值同奇偶性，就做一次 16×8 的 SAD 计算，得到最小 SAD 的点即最佳整像素匹配点和运动矢量的整像素部分。

如果不是 1/4 取样，依次去匹配 8 个可能的半像素点。分情况做 SAD，和帧的运动估计一样，只是做 SAD 时要用相同场相应的像素值。选出 8 个可能半像素点中最小的 SAD 即得到最佳匹配半像素点和半像素的运动矢量。

如果是 1/4 取样，依次去匹配 8 个可能的半像素点，将整像素的坐标值乘以 4 加上可能的半像素点乘以 2 即 1/4 精度单位的点依次做亮度块双线性内插，和以前的帧内插有所不同，这里面进行 16×8 矩阵的内插，每处理完一行参考指针要往下偏移 2 行，以便处理相同的场。最后得到一个 16×8 的预测矩阵，将这个预测矩阵与当前编码场做 SAD。当 8 个可能的半像素点都进行了内插和 SAD 处理后，得到最小 SAD 值的那个点即最佳半像素匹配点。再求 1/4 像素精度的最佳匹配点：将最佳匹配的整像素坐标值乘以 4 加上刚求得的半像素位移乘以 2 即得到最佳匹配半像素位置，将最佳匹配半像素位置加上可能的 1/4 像素点的偏移量即以 1/4 精度为单位的参考点依次做亮度块双线性内插，这里面也进行 16×8 矩阵的内插，最后得到一个 16×8 的预测矩阵，将这个预测矩阵与当前编码场做 SAD。当 8 个可能的 1/4 像素点都进行了内插和 SAD 处理后，得到最小 SAD 值的那个点即最佳 1/4 像素匹配点，而与最佳匹配半像素点的偏移量即运动矢量的分数部分。前向顶场参考顶场的 16×8 匹配完成。

接着进行当前前向顶场参考底场的 16×8 匹配，过程和前向顶场运动矢量参考底场的匹配基本一样，只是参考亮度要往下偏移一行。

如果前向顶场参考顶场得到的 SAD 小于前向顶场参考底场得到的 SAD，那么当前宏块模式中前向顶场参考底场的标志置 0；否则置 1。最小的 SAD 作为 TSAD。

接着进行前向底场参考顶场的 16×8 匹配，当前场是底场，所以预测块要偏移 16 个像素点。

接着进行前向底场参考底场的 16×8 匹配，因为是参考了底场，所以参考亮度指针要往下偏移一行，当前场是底场，所以预测块要偏移 16 个像素点。

如果前向底场参考顶场得到的 SAD 小于前向底场参考底场得到的 SAD，那么当前宏块模式中前向底场参考底场的标志置 0；否则置 1。最小的 SAD 作为 BSAD，TSAD 加上最小的 BTSAD 作为 FLDSAD。

接着依次对每个亮度块进行 8×8 的匹配：在编码块与参考帧中预测矢量和坐标值确定的位置做 8×8SAD0。然后在由预测运动矢量和当前坐标指向的位置周围进行半径为 2 螺旋搜索即 24 个点的 SAD 值，找的最小 SAD 值即最佳匹配整像素点。如果是 1/2 像素取样，逐一以最佳匹配整像素周围 8 个半像素点为顶点的 8×8 块与当前编码的块进行 SAD 计算，过程与做整像素的 SAD 一样，但是每处理完一行后扩展参考指针要往后偏移 2 个扩展亮度块的宽度。之后最小 SAD 点即最佳半像素匹配点。如果是 1/4 像素取样，于是将最佳整像素匹配点乘以 4，其周围的半像素点乘以 2 这样就是 1/4 像素表示法，进行帧间亮度块双线性内插，可以得到

一个 16×16 由参考帧经过插值形成的矩阵，然后将这个矩阵与当前编码亮度块进行 SAD 计算，依次比较 8 个半像素点找到最小的 SAD 值即最佳匹配半像素点，因为要得到 1/4 像素的运动矢量，所以要将运动矢量和参考点以 1/4 精度表示。再以最佳半像素点为中心在它周围 8 个 1/4 精度点逐一进行帧间亮度块双线性内插得到一个内插块，再对这个内插块和编码块进行 SAD 计算，当 SAD 值最小的点即 1/4 像素精度最佳匹配点，也可以得到运动矢量的小数部分，将运动矢量赋给运动矢量序列中的当前宏块第一个亮度块的运动矢量。然后用同样的方法求第二个亮度块的 SAD 和它的运动矢量，只是当前亮度的指针偏移 8 个像素点指向第二个块。并依次将 4 个亮度块的 SAD 总和为 SAD8。

在 FLDSAD、SAD、SAD8 这三个值中，如果 SAD 最小，那么此宏块类型为 INTER 只有一个 MV；否则再判断如果 FLDSAD 小于 SAD，那么依次给前向顶场和底场赋上已经求得的值，否则宏块类型为 INTER4V。这样就处理完一个宏块。

3.2.2 双向运动估计编码框架

BVOP 的运动估计：首先找到当前帧和两个参考帧的起始点，然后对当前帧的各个宏块做运动估计。先将当前宏块与前向无位移的参考宏块做一次 SAD 计算，得到 SAD0；再进行前向 16×16 匹配，这里的 16×16 匹配过程和 PVOP 中的一样，先计算整像素的 SAD 值，再求 1/2 或 1/4 像素精度的 SAD1 值，而 1/4 像素精度的搜索先进行双线性内插求出 1/2 像素精度的运动矢量，再双线性内插变成求出 1/4 精度的运动矢量，即得到前向运动矢量。

接着做后向 16×16 匹配，过程和前向 16×16 匹配一样，只是参考后面参考帧中的参考宏块，得到后向运动矢量和 SAD2。

接着用求得的前向和后向运动矢量做双线性内插后再求帧 SAD。如果是 1/4 像素取样，那么分别用前向参考帧和后向参考帧做 1/4 像素的运动补偿，如果是逐行编码则用已经求得的运动矢量对参考宏块做亮度双线性内插，分别得到了当前宏块的前向和后向预测宏块。再用当前宏块的亮度值跟，前向和后向预测宏块亮度的平均值做 SAD，得到 SAD3。

然后选择合适的 16×16 宏块类型：如果 SAD1 小于 SAD2 那么宏块类型为帧编码的 FORWARD；如果 SAD1 大于 SAD2 那么宏块类型为帧编码的 BACKWARD；如果 SAD3 加上 129 后的值比 SAD1 和 SAD2 都小，那么宏块类型为帧编码的 INTERPOLATE。

如果当前帧是隔行编码的，分别找到前向和后向参考场中参考宏块的位置，

接着选择最佳的前向顶场预测器。

进行前向顶场参考顶场的 16×8 匹配：求得的运动矢量放在前向的顶场参考顶场 FTT 中。先做当前宏块的顶场和没有位移的参考宏块顶场 16×8 的 SAD0。接着在搜索范围内做搜索，找到整像素最小的 SAD1 值和整像素运动矢量。

如果是 1/2 像素精度的运动估计，如果半像素搜索点在整像素的位置，那么直接将参考场和当前场做场的 SAD 计算；如果半像素搜索点在行的整像素位置而列在半像素位置，那么将参考场一列中相邻的两个像素值凑整右移一位和当前场相应的一个像素值做场的 SAD 计算；如果半像素搜索点在行的半像素位置而列在整像素位置，那么将参考场一行中相邻的两个像素值凑整右移一位和当前场相应的一个像素值做场的 SAD 计算；如果半像素搜索点在行和列的半像素位置，那么将参考场一行中相邻的两个像素值和下一行中相邻的两个像素值求和并凑整右移两位和当前场相应的一个像素值做场的 SAD 计算。

如果是 1/4 像素精度的运动估计，先计算半像素最佳匹配点再计算 1/4 像素最佳匹配点。为此先进行亮度块场的双线性内插，过程和帧的双线性内插一样只是得到一个 16×8 的内插块用来计算最佳半像素匹配点，是用这个 16×8 的内插块和当前块做 SAD，得到最佳半像素搜索点和最佳半像素运动矢量。再在最佳匹配半像素点周围找最佳匹配 1/4 像素精度点，还是先进行亮度块场的双线性内插，又得到一个 16×8 的内插块用来在最佳半像素点周围计算出最佳匹配 1/4 精度像素点，是用这个 16×8 的内插块和当前块做 SAD，得到最佳 1/4 像素精度搜索点和 1/4 像素精度运动矢量。最后得到的最小 SAD 为 SADFld[0]。

进行前向顶场参考底场的 16×8 匹配：求得的运动矢量放在前向的顶场参考底场 FTBMV 中，参考场为参考宏块的底场。过程和顶场参考一样。最后得到的最小 SAD 为 SADFld[1]。

如果 SADFld[0]小于 SADFld[1]，那么当前宏块模式中前向顶场参考底场的标志符置 0，并将 SADFld[0]赋给 SADFldMB[0]；否则前向顶场参考底场的标志符置 1，并将 SADFld[1]赋给 SADFldMB[0]。

接着选择最佳的前向底场预测器。

再进行前向底场参考顶场的 16×8 匹配：求得的运动矢量放在前向的底场参考顶场 FBTMV 中，参考场为前向参考宏块的顶场，而当前场为底场，过程和以上类似。最后得到的最小 SAD 为 SADFld[2]。

再进行前向底场参考底场的 16×8 匹配：求得的运动矢量放在前向的底场参考底场 FBBMV 中，参考场为前向参考宏块的底场，而当前场为底场，过程和以

上类似。最后得到的最小 SAD 为 SADFld[3]。

如果 SADFld[2]小于 SADFld[3]，那么当前宏块模式中前向底场参考底场的标志符置 0，并将 SADFld[0]赋给 SADFldMB[0]；否则前向底场参考底场的标志符置 1，并将 SADFld[1]赋给 SADFldMB[0]。

接着选择最佳的后向顶场预测器。

再进行后向顶场参考顶场的 16×8 匹配：求得的运动矢量放在后向的顶场参考顶场 BTTMV 中，参考场为后向参考宏块的顶场，而当前场为顶场，过程和以上类似。最后得到的最小 SAD 为 SADFld[4]。

再进行后向顶场参考底场的 16×8 匹配：求得的运动矢量放在后向的顶场参考底场 BTBMV 中，参考场为后向参考宏块的底场，而当前场为顶场，过程和以上一样。最后得到的最小 SAD 为 SADFld[5]。

如果 SADFld[4]小于 SADFld[5]，那么当前宏块模式中后向顶场参考底场的标志符置 0，并将 SADFld[4]赋给 SADFldMB[1]；否则后向顶场参考底场的标志符置 1，并将 SADFld[5]赋给 SADFldMB[1]。

接着选择最佳的后向底场预测器。

再进行后向底场参考顶场的 16×8 匹配：求得的运动矢量放在后向的底场参考顶场 BBTMV 中，参考场为后向参考宏块的顶场，而当前场为底场，过程和以上类似。最后得到的最小 SAD 为 SADFld[6]。

接着再进行后向底场参考底场的 16×8 匹配：求得的运动矢量放在后向的底场参考底场 BBBMV 中，参考场为后向参考宏块的底场，而当前场为底场，过程和以上类似。最后得到的最小 SAD 为 SADFld[7]。

如果 SADFld[6]小于 SADFld[7]，那么当前宏块模式中后向底场参考底场的标志符置 0，并将 SADFld[6]赋给 SADFldMB[1]；否则后向底场参考底场的标志符置 1，并将 SADFld[7]赋给 SADFldMB[1]。

选择最佳前向或后向场预测：

如果 SADFldMB[0]小于等于 SADFldMB[1]，那么将 SADFldMB[0]赋给 FieldSAD，当前宏块类型就为场编码的 FORWARD；否则 SADFldMB[1]赋给 FieldSAD，当前宏块类型就为场编码的 BACKWARD。

将 FieldSAD 加上 129 赋给 FieldSAD。

接着用求得的前向和后向运动矢量做双线性内插后再求场 SAD：如果是 1/4 像素精度取样，进行 1/4 像素的运动补偿得到当前宏块的前向顶场预测值，而当前宏块的顶场预测值是根据宏块模式中前向顶场参考底场的标志符来决定取前向参

考场的顶场还是底场来做运动补偿，以得到当前宏块的前向顶场预测值。否则进行 1/2 像素精度运动补偿。

如果是 1/4 像素精度取样，进行 1/4 像素的运动补偿得到当前宏块的前向底场预测值，而当前宏块的底场预测值是根据宏块模式中前向底场参考底场的标志符来决定取前向参考场的顶场还是底场来做运动补偿，以得到当前宏块的前向底场预测值。否则进行 1/2 像素精度运动补偿。

接着依次进行，由后向顶场的运动补偿得到当前宏块后向顶场的预测值，接着由后向底场的运动补偿得到当前宏块后向底场的预测值。然后对这个补偿得到的 16×16 数组和当前宏块像素值做 SAD：将前向预测值（包括前向顶场和前向底场）加上后向预测值（包括后向顶场和后向底场）加一往右移动一位得到的值后与当前宏块相应的亮度值做 SAD，得到 SADFieldMB[2]。

如果 SADFieldMB[2]加上 129 后小于 FieldSAD，那么将 SADFieldMB[2]加上 129 后赋给 FieldSAD，宏块类型赋为场编码的 INTERPOLATE。

有一种强制性判断：如果后向参考没编码，那么将 SADFieldMB[0]赋给 FieldSAD，而宏块类型强制性转换为场编码的 FORWARD。

再进行场和帧对比：如果 FieldSAD 小于 SAD，那么用场运动矢量做运动补偿标志置位，将 FieldSAD 赋给 SAD。

如果相应位置的宏块存在，要计算 DIRECT 模式的前向和后向运动矢量。如果参考宏块模式中用场运动矢量做运动补偿标志符为真，那么进行 DIRECT 模式中场的 SAD，否则进行 DIRECT 模式中帧的 SAD。

DIRECT 模式下帧的 SAD：如果参考宏块模式为 INTRA 或 INTRAQ，那么参考宏块的运动矢量为 0。

在 DIRECT 模式的搜索范围内水平和垂直方向都是[-2，2]即 25 个微型差分矢量对四个亮度块依次进行如下处理：将后向参考宏块中相应块的运动矢量经过前向线性缩放后得到的值加上一个微型差分矢量赋给当前块的前向运动矢量。如果这个微型差分矢量的水平分量不为 0，那么将当前块的前向运动矢量的水平分量减去后向参考宏块中相应块运动矢量的水平分量所得到的值赋给当前块的后向运动矢量的水平分量；否则将后向参考宏块中相应块运动矢量的水平分量做后向线性缩放所得到的值赋给当前块后向运动矢量的水平分量；垂直分量做类似处理。这就得到四个亮度块的前向和后向运动矢量。

然后分别对四个亮度块进行运动补偿得到前向和后向预测值。可以是 1/4 或 1/2 像素的运动补偿，每处理完一个亮度块，预测指针就要做相应的移动，处理完

后可得到一个16×16的预测宏块亮度值。接着对前向预测和后向预测亮度块的平均值跟当前亮度块的亮度值做SAD。如果求得的SAD值小于directSAD（一个缺省值）时，那么将求得的SAD值赋给directSAD，并将微型差分矢量赋给当前宏块DIRECT模式的差分运动矢量。

DIRECT模式下场的SAD：和求DIRECT模式下帧的SAD过程类似，不同点是：线性缩放时的TRB、TRD也要做一些处理；先用参考宏块顶场的运动矢量做线性缩放得到当前宏块的前向顶场运动矢量，然后用前向顶场运动矢量得到后向顶场运动矢量；再用参考宏块底场的运动矢量做线性缩放得到当前宏块的前向底场运动矢量，然后用前向底场运动矢量得到后向底场运动矢量；接着进行1/2或1/4精度的顶场前向和后向运动补偿，再进行1/2或1/4精度的底场前向和后向运动补偿，这样就得到一个16×16的前向和后向预测亮度值，将预测值的平均和当前宏块做SAD。如果求得的SAD值小于directSAD（一个缺省值）时，那么将求得的SAD值赋给directSAD，并将微型差分矢量赋给当前宏块DIRECT模式的差分运动矢量。

再选择最好的宏块类型：如果directSAD减去129不小于SAD，那么用DIRCT模式以上的各种参数值；否则宏块类型为DIRECT，当参考宏块模式中用场运动矢量做运动补偿的标志赋给当前宏块，如果这个标志为真，那么前向和后向的顶场和底场运动矢量有效。否则即帧编码的DIRECT模式，再判断是否参考宏块4M标志为真，则依次得到4个FMV和4个BMV，否则就是参考宏块只有1MV帧编码的DIRECT模式。

3.3 提高运动估计效率的主要因素

3.3.1 初始搜索点的选择

（1）直接选择参考帧的（0，0）位置。这种方法简单，但易陷入局部最优点。如果采用的算法初始步长太大，而原点又不是最优点，又可能使快速搜索跳出原点周围可能比较大的区域而去搜索远距离的点，导致搜索方向的不确定性，故有可能陷入局部最优。

（2）选择预测的起点。相邻快之间具有很强的相关性，以预测点作为搜索起点。大量的实验证明预测点更加靠近最佳匹配点，使得搜索次数减小。序列图像的运动矢量再空间和时间上具有很强的相关性，特别使属于同一对象的块运动保

持一致的可能性更大，这时若取相邻块的运动矢量来预测搜索起点，会使中心偏移更加集中。

3.3.2 块匹配的准则

运动估计算法中常用的匹配准则有 3 种，即最小绝对差（MAD）、最小均方误差（MSE）和归一化互相关函数（NCCF）。子采样匹配准则在一定的应用中也被使用。匹配准则对匹配的精度影响不是很大，所以不含乘除法的绝对差值和 SAD 准则成为最常使用的匹配准则。

$$MAD(i,j)=\frac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N}\left|f_k(m,n)-f_{k-1}(m+i,n+j)\right| \tag{3-2}$$

式(3-2)为最小绝对差（MAD）其中，(i,j) 为位移矢量，f_k 和 f_{k-1} 分别为当前帧和上一帧的像素值，M×N 为宏块的大小，若在某一个点 (i_0,j_0) 出 $MAD(i_0,j_0)$ 达到最小，则该点为要找的最优匹配点。通常使用绝对差值和 SAD 代替 MAD,即

$$SAD(i,j)=MN*MAD(i,j) \tag{3-3}$$

3.3.3 搜索策略

搜索策略选择恰当与否对运动估计的准确性和速度都有很大的影响。

1 全搜索法（Full Search Method，FS）

FS 是对搜索范围内所以可能的候选位置计算 $SAD(i,j)$ 值，从中找出最小 SAD，其对应偏移量即为所求运动矢量。此算法虽最简单、可靠，找到的必为搜索范围内的全局最优点。但计算量大[15]、难于用于实时应用，多模式的 BMME 占据了整个编码大约 80%的计算量[16]。

2 减少搜索复杂度和计算复杂度的方法

通过限制搜索位置的数目来减少计算量。如 TSS、TDL、CS、FSS、DS 等。算法固定搜索模板和步长，大大减少了搜索次数。

采用子采样方法计算块匹配失真来降低计算复杂度。通常的匹配准则是把块里所有的像素点进行计算和比较，事实上一个块相邻像素的差别很小，使得它们之间也存在冗余。子采样匹配准则就利用了这一事实，只取其中的一部分像素进行计算，可以大大减少计算量，但同时降低了准确性，故子采样匹配准则通常与多候选点方法相结合。

采用中止判别准则来减少搜索复杂度。实际中绝大多数图像序列的运动都很小，如视频电话、视频会议的图像序列，背景几乎是静止的，前景的运动也不大。

它们的运动矢量通常总是高度集中分布在搜索窗的中心位置附近。在进行初始点预测后，预测起点更接近最优点，所以没有必要对这些块全部按模板由大到小进行匹配运算，可以先利用小模板并通过一个的准则来判断初始搜索块是否就是最优匹配块，如果满足判别准则就直接结束搜索。

3 防止搜索陷入局部极小的方法

采用中心倾向的搜索点模式。在许多视频应用中，图像序列的块运动通常只包含微小的运动，绝大多数的运动矢量分布在原点附近，基于这种中心倾向的考虑，NTSS、FSS、BBDS 等算法在第一步中采用了中心倾向的搜索点模式，提高了匹配速度，减小了陷入局部极小的可能性。

采用多个候选点作为下一步搜索的中心位置。多候选点方式搜索选择多个失真度较小的点作为下一步搜索的中心位置，个数可依据搜索速度和精度进行调整。

4 基于运动内容灵活处理不同的运动块

通过一定的方法将块运动类型判别出来就可以根据图像中的运动类型，自适应选择下一步相应的搜索模板和搜索范围，对不同的运动块有不同大小的搜索模板和搜索步长，使搜索与图像的内容有关（即基于内容的搜索），从而得到较好的估值结果。该类算法（如 SDS[17]）的搜索不需要经历如 DS 算法一样的模板由大到小的必然过程，它可以充分利用运动矢量的中心偏移性，对序列图像中存在的大量静止和小运动块实现一步搜索，提高了搜索速度，大大减小了计算量。

5 提高搜索精度降低码率的方法

半像素精度运动估计。在 H.263 中使用了半像素搜索精度，这对于提高图像编码过程中小尺寸运动矢量估计精度是非常有用的。通过将在匹配位置周围领域的像素进行内插，然后进一步搜索，可以找到更精确匹配的位置。运动矢量精度的提高使经运动补偿后的帧间预测误差减小，从而降低了码率。

1/4 像素精度运动估计。对 16×16 的宏块分成 4 个 8×8 的亮度块分别进行运动估计可得到 4 个运动矢量。如果利用 4 个运动矢量所得的预测误差比使用整个宏块估值所得到的单个运动矢量时的预测误差小很多，则传送 4 个运动矢量。这对于宏块内部个块运动不一致的情况效果很好。虽然此时传送运动矢量所花费的比特数增加了一些，但由于预测误差的大幅度降低，仍然使总码率降低。并且采用 4 个块进行运动估计在补偿预测时可以采用重叠预测法消除块效应。

3.4 菱形算法的原理与实现

在视频压缩编码领域,经典运动估计的方法有很多,比如 FS(full searching) 、TSS (three step search) 、NTSS (new three step search) 、TWSS[19] (two step search) 、CSA(cross search algorithm) 、DS (diamond search)等,其中由于 DS 算法可以获得与 TSS、NTSS 相当的图像质量而搜索次数却大为下降而备受青睐。

实时视频编码系统要求运动估计的搜索算法即快又准，现行的菱形搜索算法是公认的效率较高的算法之一。经典菱形搜索步骤如下,其中初始量化步长为 2 ,衡量最佳匹配准则为 SAD。

(1) 先算出在参考帧中当前点(图 3.2 中标号为 0 的点,其坐标为(0 ,0)) 的 SAD ,然后以 0 点为中心,搜索大菱形 8 个点,也就是图 3.2 中标号为 1 和 2 点。

(2) 如果中心点(图 3.2 中标号为 0) 的 SAD 最小,则继续搜索小菱形的 4 个点,也就是图中的标号为 3 的点。这样搜索出的最小 SAD 点就认为是最匹配的点。搜索结束。

(3) 如果是菱形边上中点的 SAD 最小,比如(1 , - 1) ,或者是菱形顶点的 SAD 最小,比如(2 ,0) ,则都以最小点为中心,扩展一个大菱形。前者如图 3.3 ,后者如图 3.4 ,再执行(1)。必须注意,搜索过的点就不必再搜索。

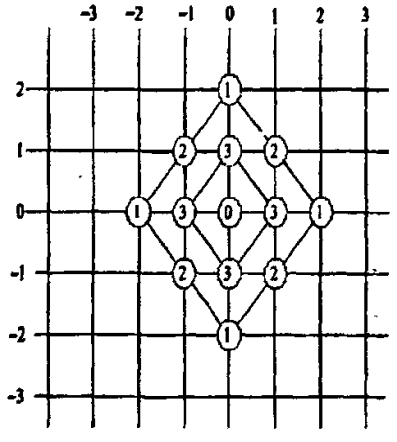

图 3.2 原始大、小菱形和中心点

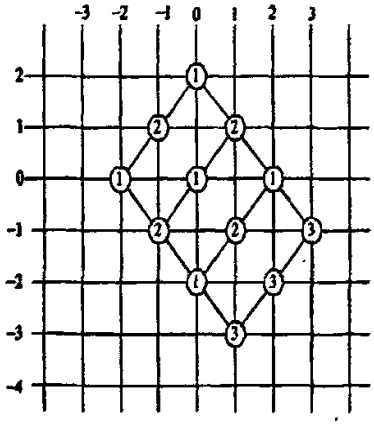

图 3.3 以大菱形边上中点为中心扩展新大菱形

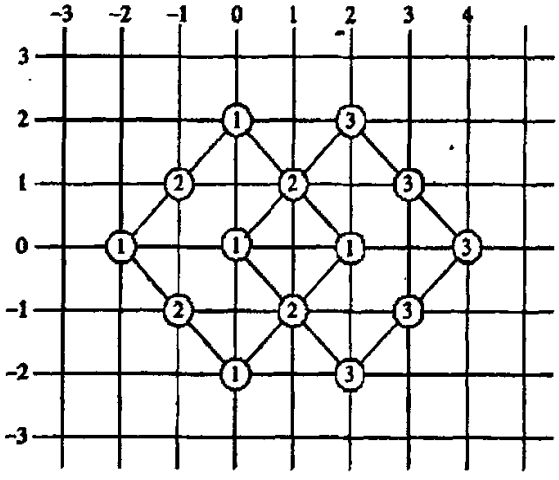

图 3.4 以大菱形顶点为中心扩展新大菱形

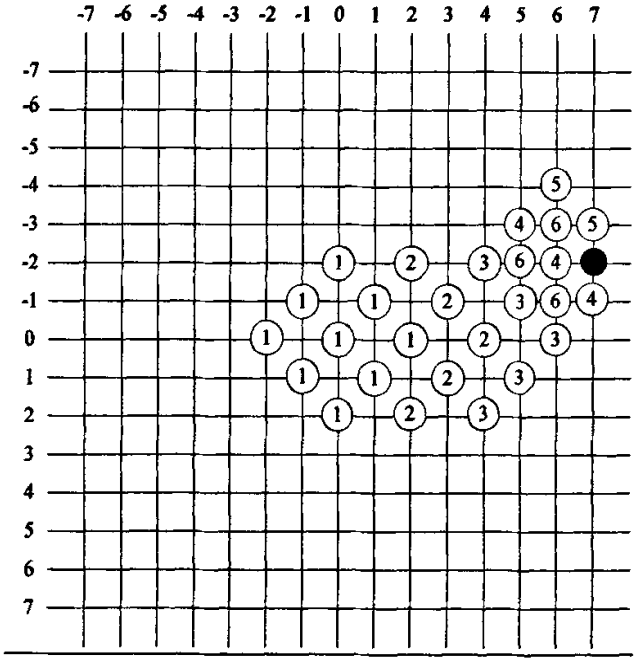

图 3.5 经典 DS 搜索路径图

典型 DS 搜索路径图：以（0，0）点为中心进行 DS 搜索，得到（2，0）为最小的 SAD 点，那么此顶点向右扩展出一个大菱形，并以此顶点为中心重新进行 DS 搜索，得到（4，0）点为最小 SAD 点，再以此顶点向右扩展一个大菱形，并重新进行 DS 搜索，得到（5，-1）即扩展大菱形边中点的 SAD 值最小，以此中点往右上角扩展一个大菱形，再进行 SAD 计算，得到（6，-2）点的 SAD 值最小，再往右上角扩展，进行 SAD 计算比较可知还是（6，-2）点的 SAD 值最小，最后

进入小菱形的SAD计算，可知（7，-2）的SAD值最小，于是认为（7，-2）为整像素最佳匹配点。

3.5 经典的菱形算法固有的缺陷

经典的菱形算法没有利用同一帧相邻宏块的运动矢量之间存在的空间相关性来预测运动矢量，可能使快速搜索跳出原点周围可能比较大的区域而去搜索远距离的点，导致搜索方向的不确定性，故有可能陷入局部最优，可对菱形搜索算法的搜索策略加以优化，在保证图像质量的同时进一步减少搜索点数。

3.6 趋势菱形算法的原理与实现

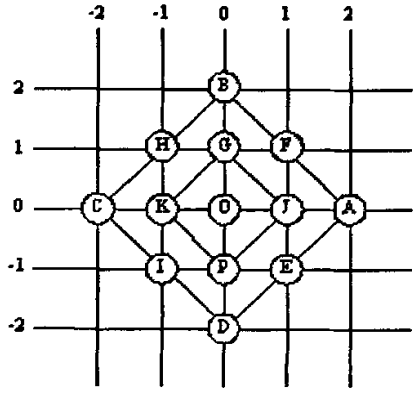

图 3.6 原始大、小菱形和中心点

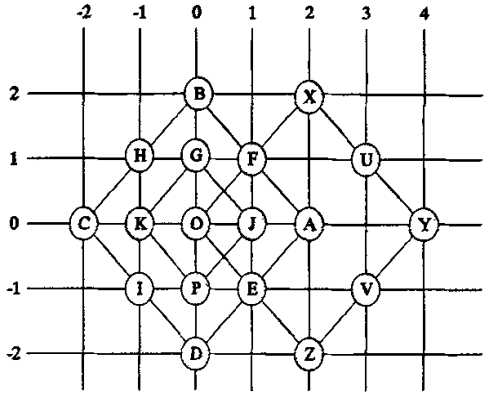

图 3.7 以大菱形顶点为中心扩展新大菱形

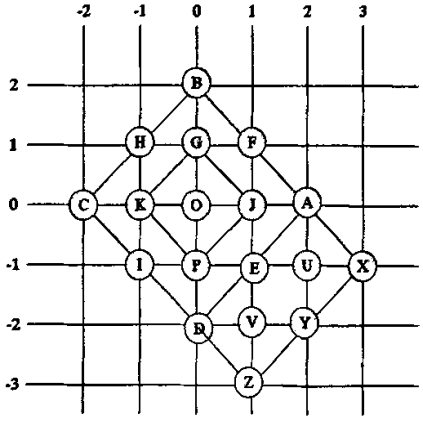

图 3.8 以大菱形边上中点为中心扩展新大菱形

1) 算出预测搜索起点（图 3.6 中标号为 O）的 SAD,然后以预测搜索起点为中心，搜索大菱形的 4 个顶点（即图 3.6 中 A,B,C,D 点）。
2) 如果中心点（图 3.6 中标号为 O）的 SAD 最小，则继续搜索大菱形边上的 4 个中点（即图 3.6 中 E,F,H,I 点）。否则跳到（4）处执行。
3) 如果仍然是中心点（图 3.6 中标号为 O）的 SAD 最小，则再继续搜索小菱形的 4 个点（图 3.6 中标号为 J,G,K,P 的点）。比较小菱形中 4 个顶点和 O 点，最小 SAD 点就认为是最匹配的点，这时搜索结束。如果不是中心点最小，比如 E 点的 SAD 最小，则以 E 为中心点，扩展出一个大菱形，如图 3.8，然后跳到（6）出执行。
4) 如果是菱形顶点上的点 SAD 最小，比如说 A,则需要比较 B，O，D 的 SAD，找出一个次最小的 SAD。如果中心点 O 的 SAD 为次最小点，则需要计算 J 点的 SAD 值，再比较 A 点和 J 点的 SAD 值，如果 J 点的 SAD 值比 A 点的 SAD 值小，则以 J 点为中心进入小菱形模式，再比较 A,F,O,E,J 点的 SAD 值，其中 O,J,A 的 SAD 值已知，只求 F,E 的 SAD 值，最小值点即为所求点，搜索结束；如果 J 点的 SAD 值比 A 点的 SAD 值大那么以 A 点为中心点，扩展一个大菱形，如图 3.7，然后执行（5）；如果 D 点（或 B 点）为 SAD 的次最小点，那么计算 E 点（或 F 点）的 SAD，如果 E 点（或 F 点）的 SAD 值比 A 点的 SAD 值大那么以 A 点为中心点，扩展一个大菱形，如图图 3.7，然后执行（5）；如果 E 点（或 F 点）的 SAD 值比 A 点的 SAD 值小，那么以 E 点（或 F 点）为中心点，扩展一个大菱形，如图 3.8，然后执行（6）；
5) 在图 3.7 中，先搜索扩展出来的大菱形的顶点，即只需搜索 3 个顶点。如果扩展出的新菱形中有些点也属于原菱形的点，那么即使上次搜索中没有搜索过点，也不再对它进行搜索。然后执行（2），如果最小 SAD 点是中心点，这时继续搜索的大菱形的边上的点只有 2 个（U 和 V）。
6) 这时扩展的大菱形只有 3 点（两个顶点和这两个顶点连线上的边上的点即 X、Z）要搜索。先比较 E、X、Z 的 SAD 值，如果如果是中心点（E）最小，则需要比较 E 和 Y 点的 SAD，如果是 E 点的 SAD 值最小，则搜索中心点附近小菱形的 4 个点(J,P,V,U)，只搜索 U、V 点，E 和这两个点中，最小 SAD 点就是最匹配点，搜索结束；如果 Y 点的 SAD 值最小，以它为中心，扩展大菱形，跳到（6）处执行。如果是顶点（Z 或 X）最小，跳到（5）处执行。

3.7 趋势菱形算法的正确性及性能比较

3.7.1 小运动矢量图像序列情况

Beijing Weather Girl 是 360×288 的小运动矢量图像序列。分别采用 FS、DS、TDS 搜索算法进行处理，得到的性能参数表、搜索点数和峰值信噪比的曲线图如表 3-1、图 3.9 所示。

表 3-1 FS、DS、TDS 搜索算法的性能参数

性能 / 图像	FS 搜索次数/MB	DS 搜索次数/MB	TDS 搜索次数/MB	FS PSNR(dB)	DS PSNR(dB)	TDS PSNR(dB)
Beijing Weather Girl	225	12.50304	11.95722	40.13671	40.08182	39.99273

针对此图像序列，TDS 算法相对 DS 算法每帧平均减小 225.97 次 SAD 计算，即搜索次数减小 4.36551%。就 PSNR 而言，DS 比 FS 算法得到的 PSNR 减小 0.05489dB，TDS 比 FS 算法得到的 PSNR 减小 0.14398dB, TDS 比 DS 算法得到的 PSNR 减小 0.08909dB。

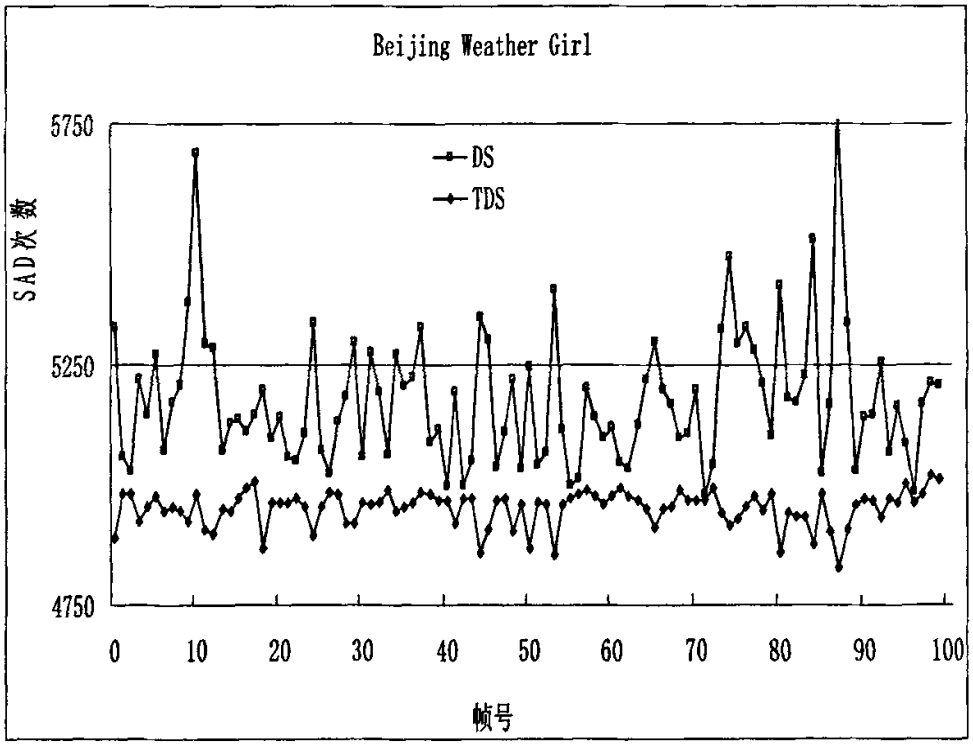

图 3.9 (a) 搜索点数比较

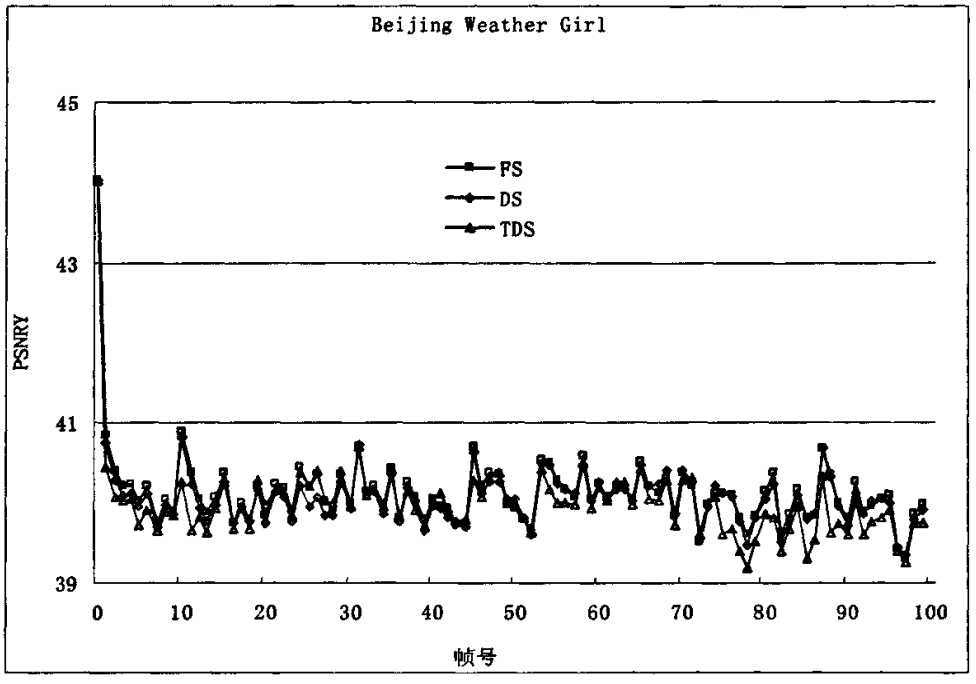

图 3.9 (b) 峰值信噪比比较

3.7.2 中等运动矢量图像序列情况

Window Car 是中等运动矢量图像序列。分别采用 FS、DS、TDS 搜索算法进行处理，得到的性能参数表、搜索点数和峰值信噪比的曲线图如表 3-2、图 3.10 所示。

表 3-2 FS、DS、TDS 搜索算法的性能参数

性能 / 图像	FS 搜索次数/MB	DS 搜索次数/MB	TDS 搜索次数/MB	FS PSNR(dB)	DS PSNR(dB)	TDS PSNR(dB)
Window Car	225	16.70768	12.32528	35.49825	35.47906	35.10427

针对此图像序列，TDS 比 DS 算法平均每帧减小 1735.43 次 SAD 计算，搜索次数减小 26.22985%。就 PSNR 而言， DS 比 FS 算法减小 0.01919dB，TDS 比 FS 算法减小 0.39398dB, TDS 比 DS 算法减小 0.37479dB。

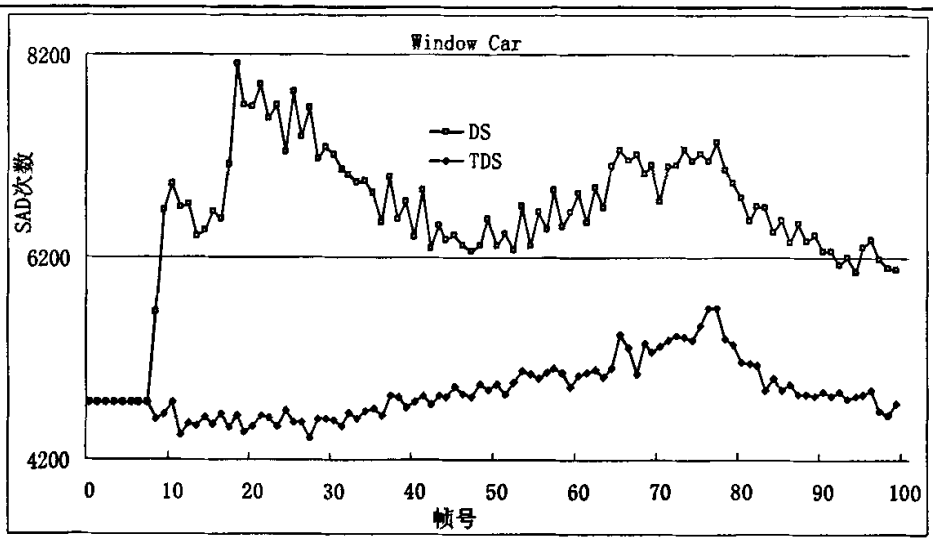

图 3.10 (a) 搜索点数比较

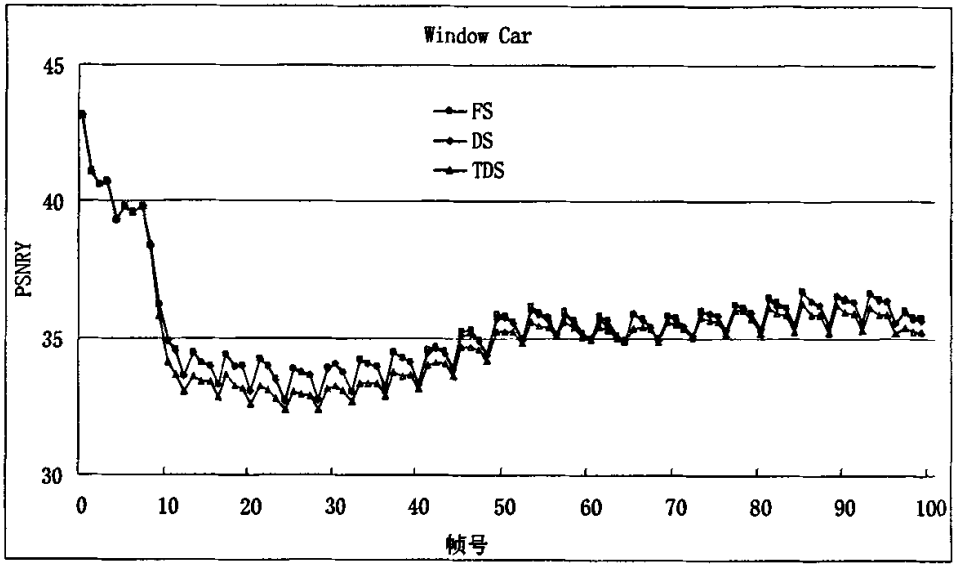

图 3.10 (b) 峰值信噪比比较

3.7.3 大运动矢量图像序列情况

Mobile Hands 是 176×144 大运动矢量图像序列。分别采用 FS、DS、TDS 搜索算法进行处理，得到的性能参数表、搜索点数和峰值信噪比的曲线图如表 3-3、图 3.11 所示。

`表 3-3 FS、DS、TDS 搜索算法的性能参数

性能 / 图像	FS 搜索次数/MB	DS 搜索次数/MB	TDS 搜索次数/MB	FS PSNR(dB)	DS PSNR(dB)	TDS PSNR(dB)
Mobile Hands	225	22.81535	13.22020	37.72408	37.68355	37.68011

针对此图像序列，TDS 算法相对 DS 算法每帧平均减小 949.92 次 SAD 计算，搜索次数减小 42.05568%。就 PSNR 而言，DS 比 FS 算法得到的 PSNR 减小 0.04053dB，TDS 比 FS 算法得到的 PSNR 减小 0.04397dB, TDS 比 DS 算法得到的 PSNR 减小 0.00344dB。

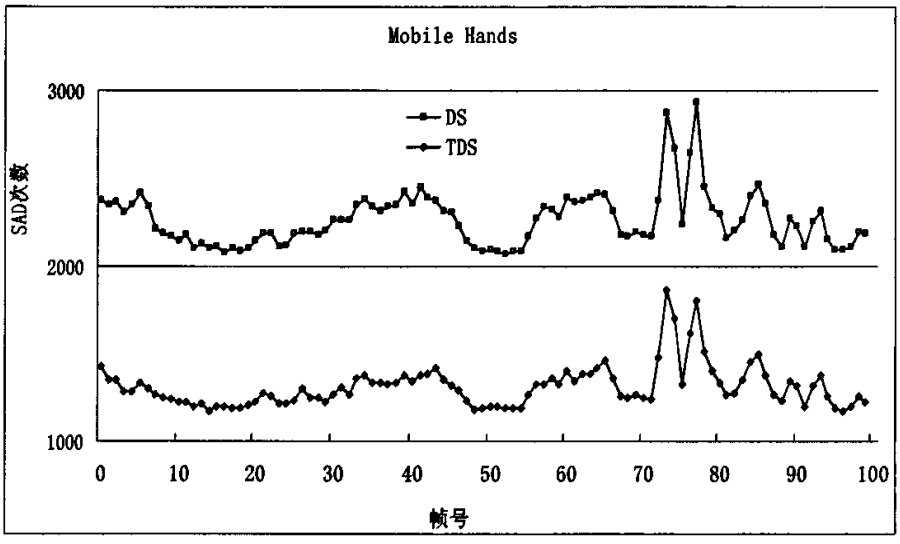

图 3.11 (a) 搜索点数比较

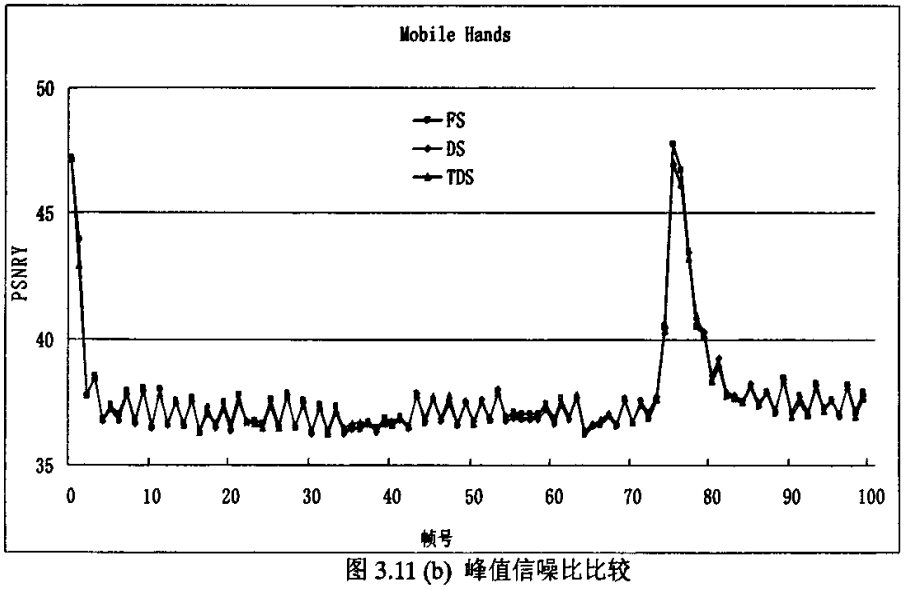

图 3.11 (b) 峰值信噪比比较

从以上三种不同运动矢量图象序列所取得的数据表明：TDS 算法利用同一帧相邻宏块的运动矢量之间存在的空间相关性来预测运动矢量，加强中心偏置特性，并对搜索策略加以优化减少了计算量，在获得与 FS 和 DS 算法相当的图像质量和信噪比的情况下，该算法有效减少了搜索次数，提高了搜索效率，尤其对中大运动矢量的视频图像运动估计效果更为明显。

3.8 形状自适应算法的原理与实现

1、算出预测搜索起点（图 3.12 中标号为 0）的 SAD, 如果其 SAD 值不大于 512，那么就停止搜索认为预测搜索起点为最佳匹配点；否则以预测搜索起点为中心，进入小菱形及其初级扩展模式 先搜索小菱形的 4 个顶点（即图 3.12 中 A, B, C, D 点）。如果中心点 0 的 SAD 最小，则认为 0 点为最佳整像素匹配点，搜索结束；如果不是则跳到（2）处执行。

2、如果是小菱形中一个顶点（如 A 点）的 SAD 最小，则比较小菱形中其它 4 个点（如 B, C, D, 0 点）的 SAD 值：如果次最小点为 0 或 C 点，则向右扩展一个小菱形，在这个扩展的小菱形中只计算 X，Y，Z 的 SAD 值，再比较 A, , X，Y，Z 的 SAD 值，如果 A 点的 SAD 值最小，则认为 A 点为最佳整像素匹配点，搜索结束；如果是 X 点的 SAD 值最小，则进入小菱形 X 方向终极扩展模式 即跳到（3）处执行；如果是 Y 点的 SAD 值最小，则进入小菱形 Y 方向终极扩展模式即跳到（4）处执行；如果是 Z 点的 SAD 值最小，则进入小菱形 Z 方向终极扩展模式即跳到（5）处执行。如果是小菱形中一个顶点（如 A 点）的 SAD 最小，而 B 或者 D 点的 SAD 值为次最小点，则进入小菱形初级对角线扩展模式即跳到（6）处执行。

3、小菱形 X 方向终极扩展模式（如图 3.13 所示）：根据 X 与 A 的位置关系再沿不同方向扩展一个小菱形，在这种情况下只计算 Y，Z 两点的 SAD，再比较 Y，Z 和扩展菱形中心点的 SAD 值。

如果扩展菱形中心点的 SAD 值最小则认为这个中心点为最佳整像素匹配点搜索结束；

如果 Y 点的 SAD 值最小则进入小菱形 Y 方向终极扩展模式 即跳到（4）处执行；

如果 Z 点的 SAD 值最小则进入小菱形 Z 方向终极扩展模式即跳到（5）处执行；

4、小菱形 Y 方向终极扩展模式（如图 3.14 所示）：根据 Y 与 A 的位置关系再沿不同方向扩展一个小菱形，在这种情况下计算 X，Y，Z 两点的 SAD。再比较 X，Y，Z 和扩展菱形中心点的 SAD 值。

如果扩展菱形中心点的 SAD 值最小则认为这个中心点为最佳整像素匹配点搜索结束；

如果 Y 点的 SAD 值最小则进入小菱形 Y 方向终极扩展模式 即跳到（4）处执行；

如果 Z 点的 SAD 值最小则进入小菱形 Z 方向终极扩展模式即跳到（5）处

执行；

如果 X 点的 SAD 值最小则进入小菱形 X 方向终极扩展模式即跳到（3）处执行；

5、小菱形 Z 方向终极扩展模式（如图 3.15 所示）：根据 Z 与 A 的位置关系再沿不同方向扩展一个小菱形，在这种情况下只计算 X，Y 两点的 SAD。再比较 X，Y 和扩展菱形中心点的 SAD 值。

如果扩展菱形中心点的 SAD 值最小则认为这个中心点为最佳整像素匹配点搜索结束；

如果 Y 点的 SAD 值最小则进入小菱形 Y 方向终极扩展模式 即跳到（4）处执行；

如果 X 点的 SAD 值最小则进入小菱形 Z 方向终极扩展模式即跳到（3）处执行；

6、小菱形对角线初级扩展模式（如图 3.16 所示），只计算 X，Y 点的 SAD 值，再比较 A，X 和 Y 点的 SAD 值。

如果 A 点的 SAD 值最小，那么认为 A 点为最佳整像素匹配点且搜索结束；

如果 X 或者 Y 点的 SAD 值最小，则进入小菱形对角线终级扩展模式即跳到（7）处执行。

7、小菱形对角线终级扩展模式（如图 3.17 所示），根据不同的方向扩展出 A，B 两点并只计算这两点的 SAD 值，再比较 A，B 两点和相应的顶点的 SAD 值。

如果相应的中心点（如 X 点）的 SAD 值最小，那么认为相应的中心点为最佳整像素匹配点且搜索结束；

如果 A 或者 B 点的 SAD 值最小，则根据不同的方向跳到（7）处执行。

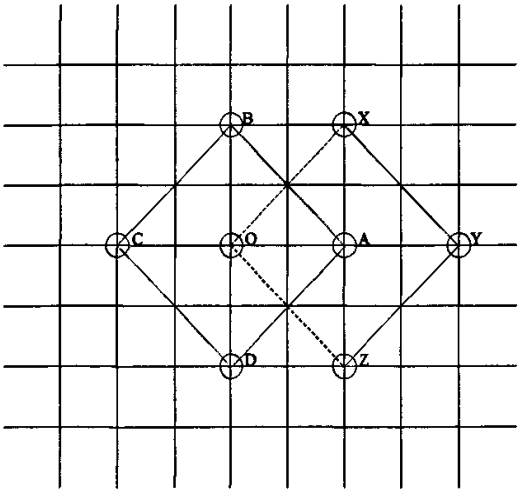

图 3.12 小菱形及其初级扩展模式

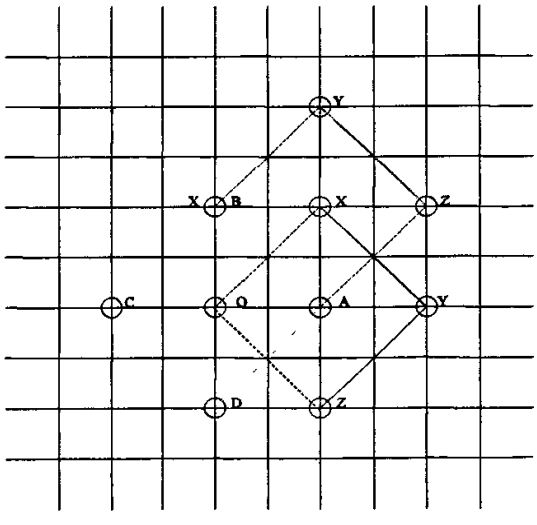

图 3.13 小菱形 X 方向终极扩展模式

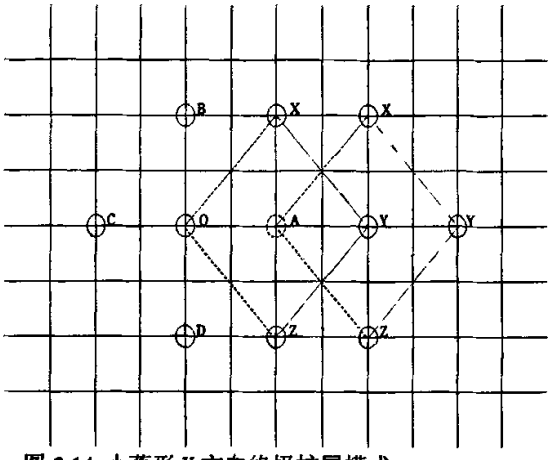

图 3.14 小菱形 Y 方向终极扩展模式

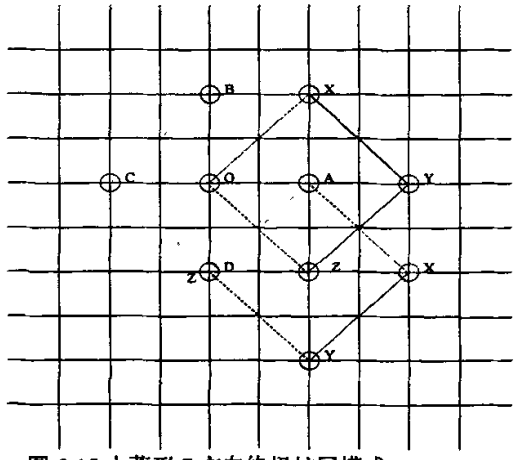

图 3.15 小菱形 Z 方向终极扩展模式

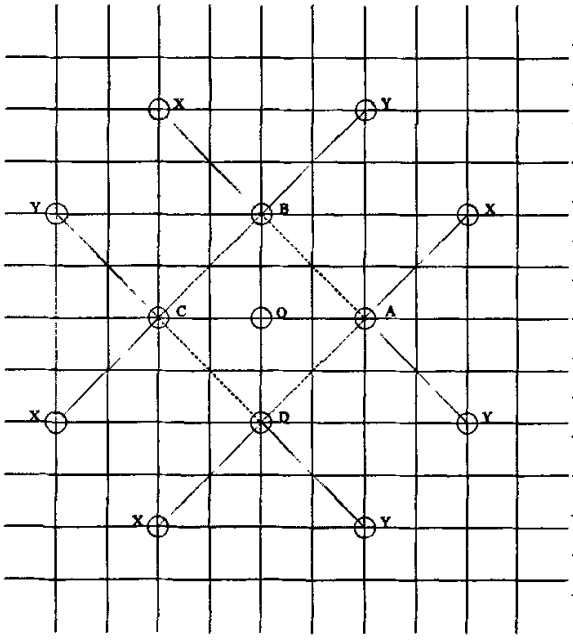

图 3.16 小菱形及对角线初级扩展模式

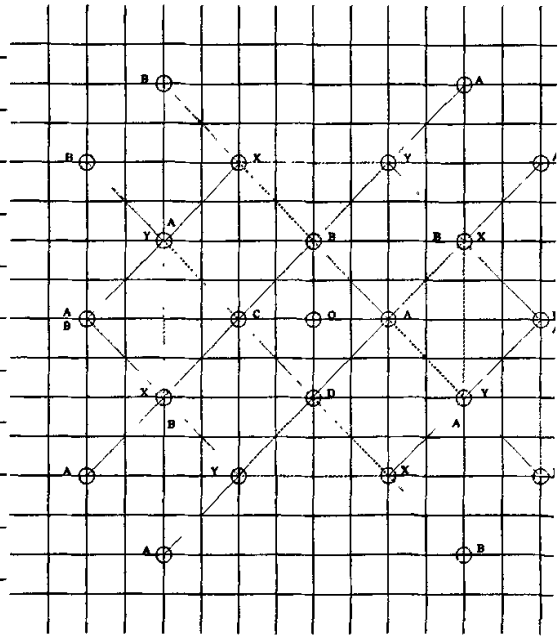

图 3.17 对角线终级扩展模式

3.9 趋势菱形算法的正确性及性能比较

3.9.1 小运动矢量图像序列情况

Beijing Weather Girl 是 360×288 的小运动矢量图像序列。分别采用 FS、DS、TDS、SAS 搜索算法进行处理，得到的性能参数表、搜索点数和峰值信噪比的曲

线图分别如表 3-4、图 3.18 所示。

表 3-4 FS、DS、TDS、SAS 搜索算法的性能参数

性能 / 图像	DS 搜索次数/帧	TDS 搜索次数/帧	TDS比DS减少搜索次数/帧	TDS比DS运算减少量	SAS 搜索次数/帧	SAS比DS减少搜索次数/帧	SAS比DS运算减少量
Beijing Weather Girl	5176.26	4950.29	225.97	4.36551%	505.34	4670.92	90.23735%

性能 / 图像	DS比FS减少（dB）	TDS比FS减少（dB）	TDS比DS减少（dB）	SAS比FS减少（dB）	SAS比DS减少（dB）	SAS比TDS减少（dB）
Beijing Weather Girl	-0.05489	-0.14398	-0.08909	-0.19302	-0.13813	-0.04904

性能 / 图像	DS 搜索次数/MB	TDS 搜索次数/MB	SAS 搜索次数/MB	FS PSNRY（dB）	DS PSNRY（dB）	TDS PSNRY（dB）	SAS PSNRY（dB）
Beijing Weather Girl	12.50304	11.95722	1.22063	40.13671	40.08182	39.99273	39.94369

针对此图像序列，SAS 算法相对 DS 算法每帧平均减小 4670.92 次 SAD 计算，即搜索次数减小 90.23735%。就 PSNR 而言，SAS 比 FS 算法得到的 PSNR 减小 0.19302dB，SAS 比 DS 算法得到的 PSNR 减小 0.13813dB。

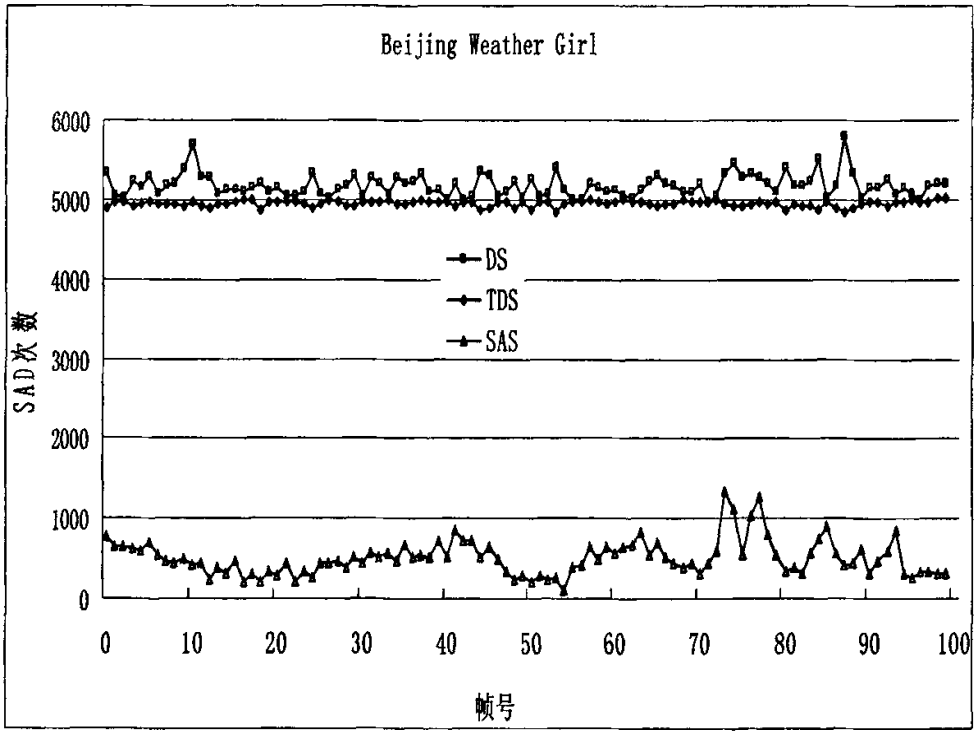

图 3.18 (a) 搜索点数比较

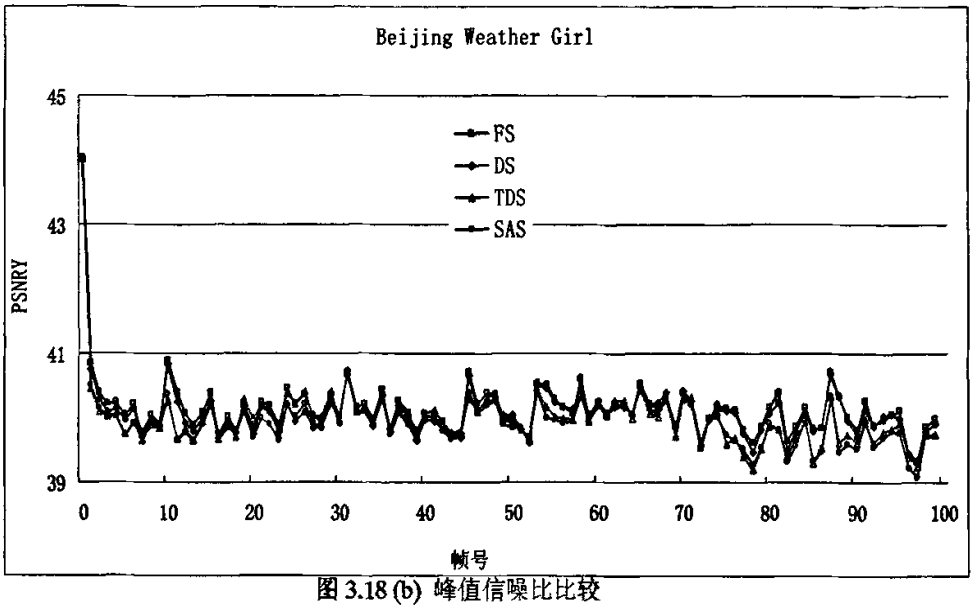

图 3.18 (b) 峰值信噪比比较

3.9.2 中等运动矢量图像序列情况

Circus 是 352×288 中等运动矢量图像序列。分别采用 FS、DS、TDS、SAS 索算法进行处理，得到的性能参数表、搜索点数和峰值信噪比的曲线图分别如表 3-5、图 3.19 所示。

表 3-5 FS、DS、TDS 搜索算法的性能参数

性能 / 图像	DS 搜索次数/帧	TDS 搜索次数/帧	TDS比DS减少 搜索次数/帧	TDS比DS 运算减少量	SAS 搜索次数/帧	SAS比DS减少 搜索次数/帧	SAS比DS 运算减少量
Circus	6515.99	4942.71	1573.28	24.14491%	2326.91	4189.08	64.28923%

性能 / 图像	DS比FS 减少（dB）	TDS比FS 减少（dB）	TDS比DS 减少（dB）	SAS比FS 减少（dB）	SAS比DS 减少（dB）	SAS比TDS 减少（dB）
Circus	-0.05589	-0.08046	-0.02457	-0.08523	-0.02934	-0.00477

性能 / 图像	DS 搜索次数/MB	TDS 搜索次数/MB	SAS 搜索次数/MB	FS PSNRY（dB）	DS PSNRY（dB）	TDS PSNRY（dB）	SAS PSNRY（dB）
Circus	16.45452	12.48159	5.87604	36.16275	36.10686	36.08229	36.07752

针对此图像序列，SAS 算法相对 DS 算法每帧平均减小 4189.08 次 SAD 计算，即搜索次数减小 64.28923%。就 PSNR 而言，SAS 比 FS 算法得到的 PSNR 减小 0.08523dB，SAS 比 DS 算法得到的 PSNR 减小 0.02934dB。

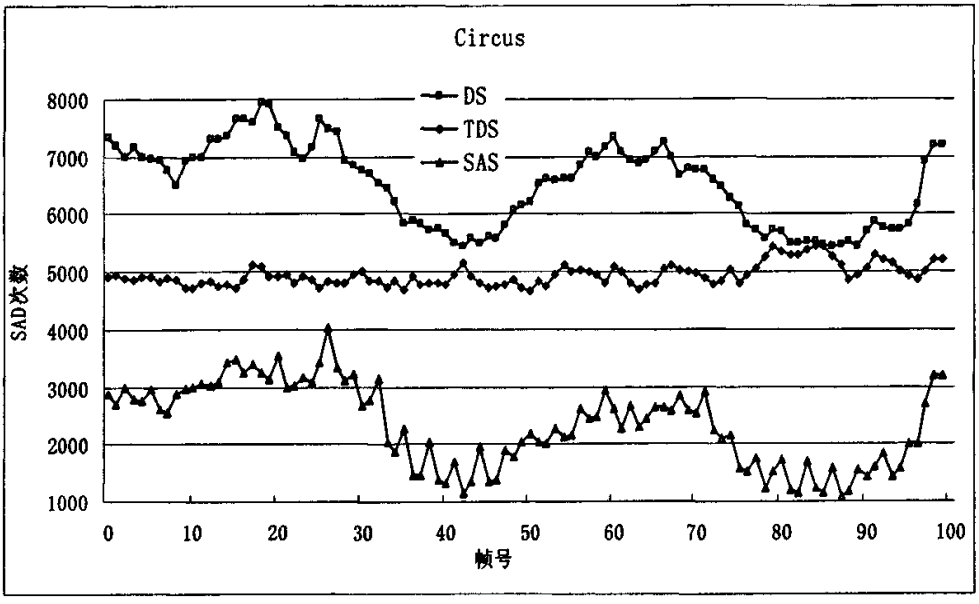

图 3.19 (a) 搜索点数比较

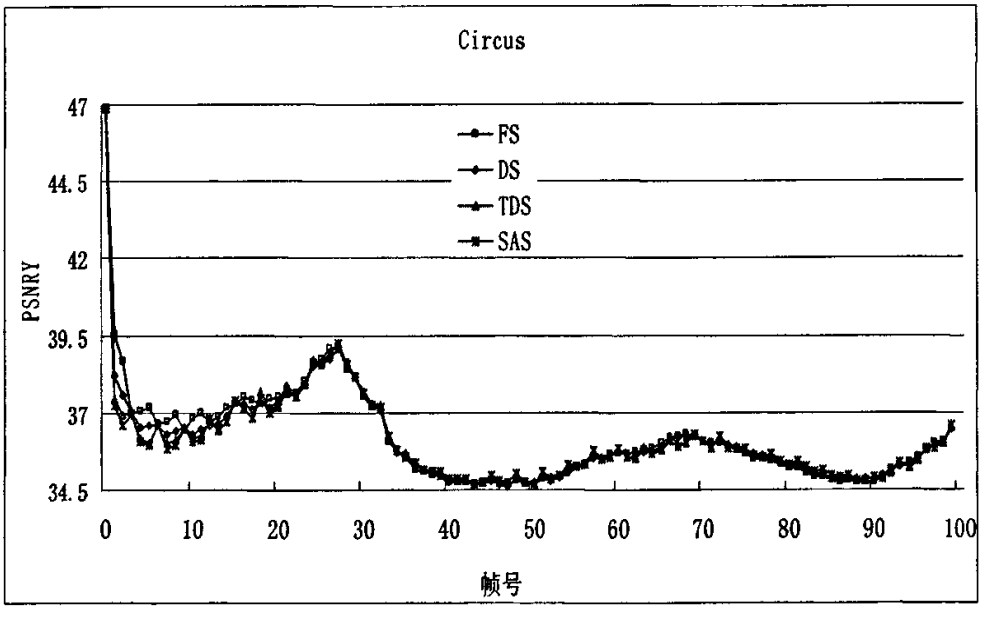

图 3.19 (b) 峰值信噪比比较

Window Car 是 352×288 中等运动矢量图像序列。分别采用 FS、DS、TDS、SAS 搜索算法进行处理，得到的性能参数表、搜索点数和峰值信噪比的曲线图分别如表 3-6、图 3.20 所示。

表 3-6 FS、DS、TDS 搜索算法的性能参数

性能 / 图像	DS 搜索次数/帧	TDS 搜索次数/帧	TDS比DS减少 搜索次数/帧	TDS比DS 运算减少量	SAS 搜索次数/帧	SAS比DS减少 搜索次数/帧	SAS比DS 运算减少量
Window Car	6616.24	4880.81	1735.43	26.22985%	2627.70	3988.54	60.28409%

性能 / 图像	DS 搜索次数/MB	TDS 搜索次数/MB	SAS 搜索次数/MB	FS PSNRY（dB）	DS PSNRY（dB）	TDS PSNRY（dB）	SAS PSNRY（dB）
Window Car	16.70768	12.32578	6.63561	35.49825	35.47906	35.10427	35.14473

性能 / 图像	DS比FS 减少（dB）	TDS比FS 减少（dB）	TDS比DS 减少（dB）	SAS比FS 减少（dB）	SAS比DS 减少（dB）	SAS比TDS 减少（dB）
Window Car	-0.01919	-0.39398	-0.37479	-0.35352	-0.33433	+0.04046

针对此图像序列，SAS 算法相对 DS 算法每帧平均减小 3988.54 次 SAD 计算，即搜索次数减小 60.28409%。就 PSNR 而言，SAS 比 FS 算法得到的 PSNR 减小 0.35352dB，SAS 比 DS 算法得到的 PSNR 减小 0.33433dB。

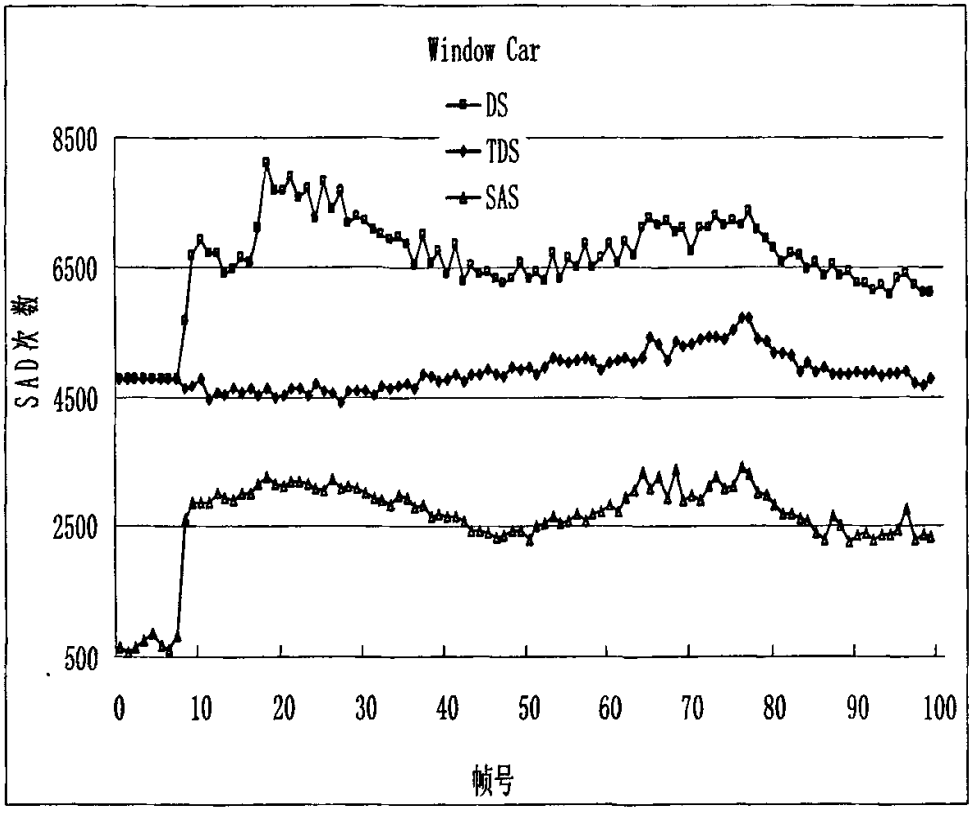

图 3.20 (a) 搜索点数比

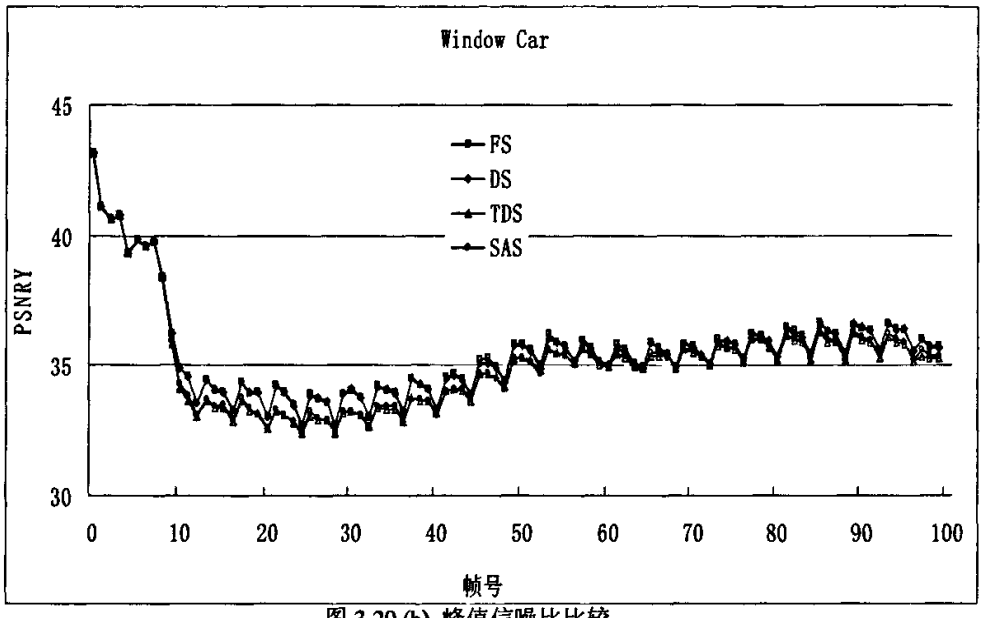

图 3.20 (b) 峰值信噪比比较

3.9.3 大运动矢量图像序列情况

Mobile Hands 是 176×144 大运动矢量图像序列。分别采用 FS、DS、TDS、SAS 搜索算法进行处理，得到的性能参数表、搜索点数和峰值信噪比的曲线图分别如表 3-7、图 3.21 所示。

表 3-7 FS、DS、TDS 搜索算法的性能参数

性能 / 图像	DS 搜索次数/帧	TDS 搜索次数/帧	TDS比DS减少搜索次数/帧	TDS比DS运算减少量	SAS 搜索次数/帧	SAS比DS减少搜索次数/帧	SAS比DS运算减少量
Mobile Hands	2258.72	1308.80	949.92	42.05568%	505.26	1753.46	77.63069%

性能 / 图像	DS比FS减少（dB）	TDS比FS减少（dB）	TDS比DS减少（dB）	SAS比FS减少（dB）	SAS比DS减少（dB）	SAS比TDS减少（dB）
Mobile Hands	-0.04053	-0.04397	-0.00344	+0.00523	+0.04576	+0.04920

性能 / 图像	DS 搜索次数/MB	TDS 搜索次数/MB	SAS 搜索次数/MB	FS PSNRY（dB）	DS PSNRY（dB）	TDS PSNRY（dB）	SAS PSNRY（dB）
Mobile Hands	22.81535	13.22020	5.10354	37.72408	37.68355	37.68011	37.72931

针对此图像序列，SAS 算法相对 DS 算法每帧平均减小 1753.46 次 SAD 计算，即搜索次数减小 77.63069%。就 PSNR 而言，SAS 比 FS 算法得到的 PSNR 增加

0.00523dB，SAS 比 DS 算法得到的 PSNR 增加 0.04576dB。

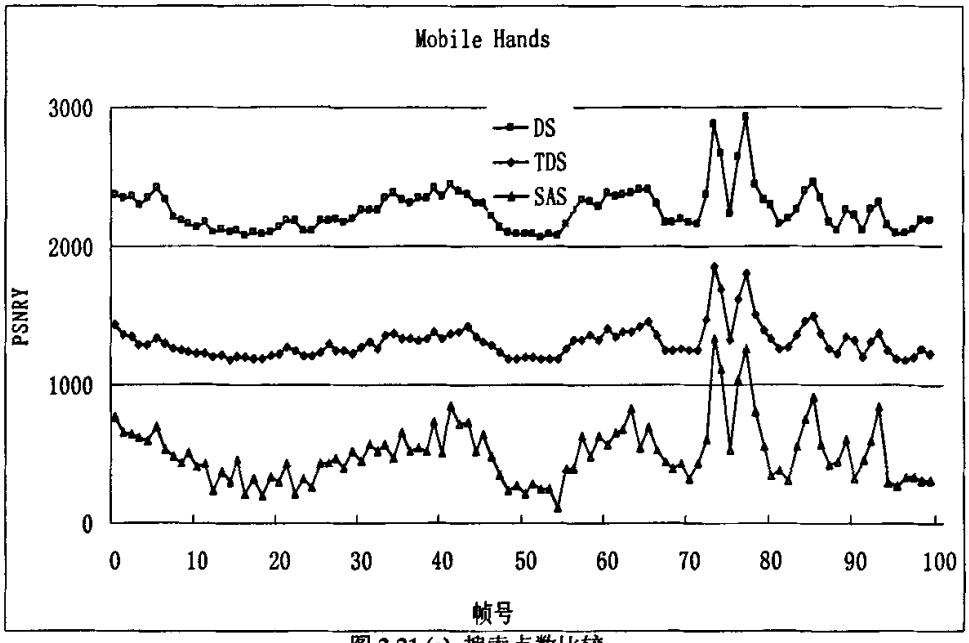

图 3.21 (a) 搜索点数比较

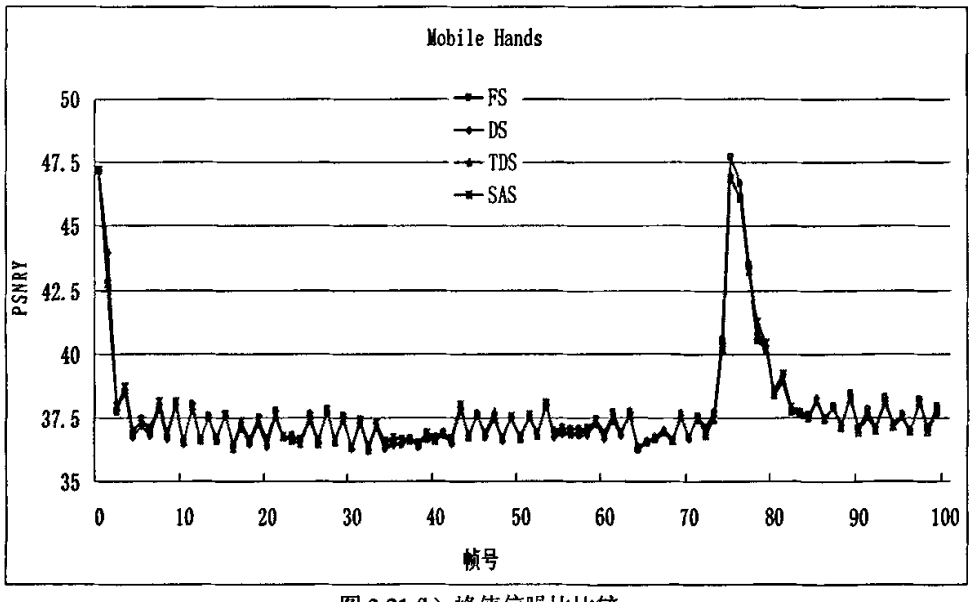

图 3.21 (b) 峰值信噪比比较

3.10 新型半像素算法的原理与实现

3.10.1 半像素全搜索算法固有的缺陷

国际标准 H.261 采用的是整像素的块匹配算法，显而易见，整像素搜索并不能代表真正的运动矢量。这是因为视频序列中帧间的位移与取样的点阵是完全无关的，若使用整像素作单位进行描述，将产生很大误差，若使用亚像素搜索可以对整像素搜索作精度上的补充，从而有效地降低很大部分的残差数据。然而快速搜索算法主要集中在整像素搜索上，半像素像素精度的搜索通常采用全搜索方式。许多学者对快速整像素运动估计算法的设计进行了大量研究，提出了整像素运动估计快速算法，当搜索区域的大小为 15×15 时，一些新兴的快速算法对每块不超过 10 个搜索点的搜索过程后就可以运算得出整像素最佳匹配点[21~24]。因此半像素全搜索中包含的线性插值运算和 8 个半像素的块失真准则运算在整个运动估计中所占的运算量已经不容忽视。

3.10.2 新新型半像素算法的原理与实现

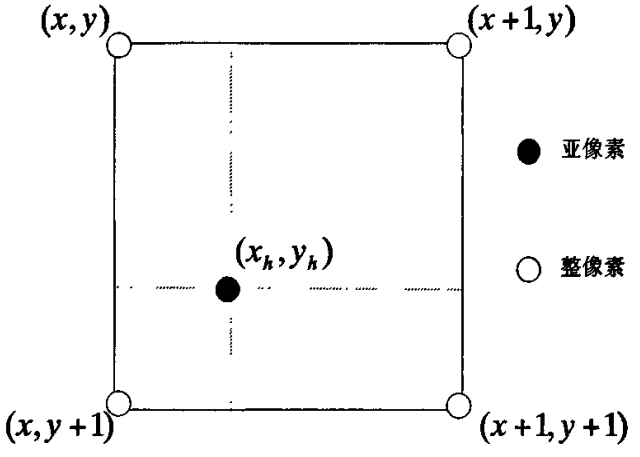

图 3.22 亚像素线性插值示意图

如图 3.22 所示，在整像素搜索结果的[-1,+1]像素领域内做亚像素线性插值，即

$$I(x_h,y_h)=I(x,y)(x+1-x_h)(y+1-y_h)+I(x+1,y)(x_h-x)(y+1-y_h)+$$
$$I(x,y+1)(x+1-x_h)(y_h-y)+I(x+1,y+1)(x_h-x)(y_h-y)$$
$$(x<x_h<x+1;y<y_h<y+1) \tag{3-4}$$

上式中，$I(x,y)$表示图像在整像素点(x,y)处的亮度值，(x_h,x_h)为亚像素点的坐标。在当前块经插值计算得到的[-1,1]像素领域内的亚像素，对每个亚像素分别计算

SAD 即可得到 SAD 误差曲面。通过观察典型的 SAD 误差曲面可以看出，由于中心整像素点的局部最小特性，领域内 SAD 误差曲面呈明显的单谷状。可以得到如下结论：在[-1,+1]像素领域内，SAD 函数随着与亚像素全局最小点距离的增加而单调增大。本论文在此基础上提出了一种新型的快速半像素运动估计算法——趋势半像素搜索法 THS，把平均搜索点数降低到 2，而重建图像的质量接近于现有的一些快速半像素运动估计算法。 原理示意如图 3.23。

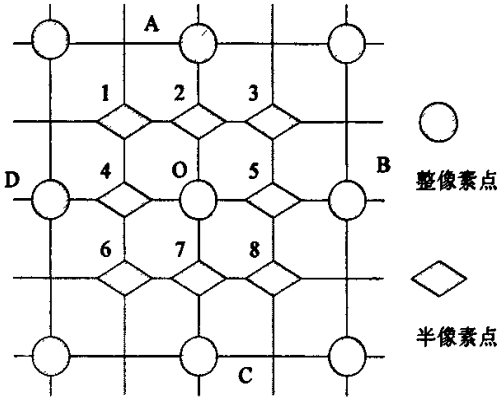

图 3.23 半像素和整像素的位置

实现原理：在得到的整像素最佳匹配点 O 后，O 点周围有 8 个半像素点和 8 个整像素点，整像素的运动估计最后搜索步长为 1，如果采用菱形搜索法或趋势菱形搜索法得到最佳匹配整像素点，那么整像素周围 4 个点(A、B、C、D)的 SAD 值是已知的，在这周围 4 个点中找出最小的 SAD 点和次最小的 SAD 点，现在只需要计算出整像素的最小 SAD 点、次最小 SAD 点与最佳匹配整像素点之间的半像素点的 SAD 值，最后再比较最佳整像素点和两个半像素点的 SAD 值，最小的为半像素匹配点。趋势半像素搜索法只需要搜索 2 个点，且算法简单。

3.10.3 半像素运动估计中趋势半像素搜索法的正确性与性能比较

Space 是 352×288 小运动矢量图像序列。分别采用半像素 FS 和 THS 搜索算法进行处理，得到的性能参数表和峰值信噪比的曲线图如表 3-8、图 3.24 所示。

表 3-8 FS 和 THS 搜索算法的性能参数

性能 / 图像	FS 搜索次数/MB	THS 搜索次数/MB	THS比FS 运算减少量	FS PSNRY（dB）	THS PSNRY（dB）	THS比FS 减少（dB）
Space	8.00000	1.97528	75.309%	43.92840	43.79259	0.13581

针对此图像序列，THS 算法相对 FS 算法每帧平均减小 2385.79 次 SAD 计算，

搜索次数减小 75.309%。就 PSNR 而言，THS 比 FS 算法得到的 PSNR 减小 0.13581dB。

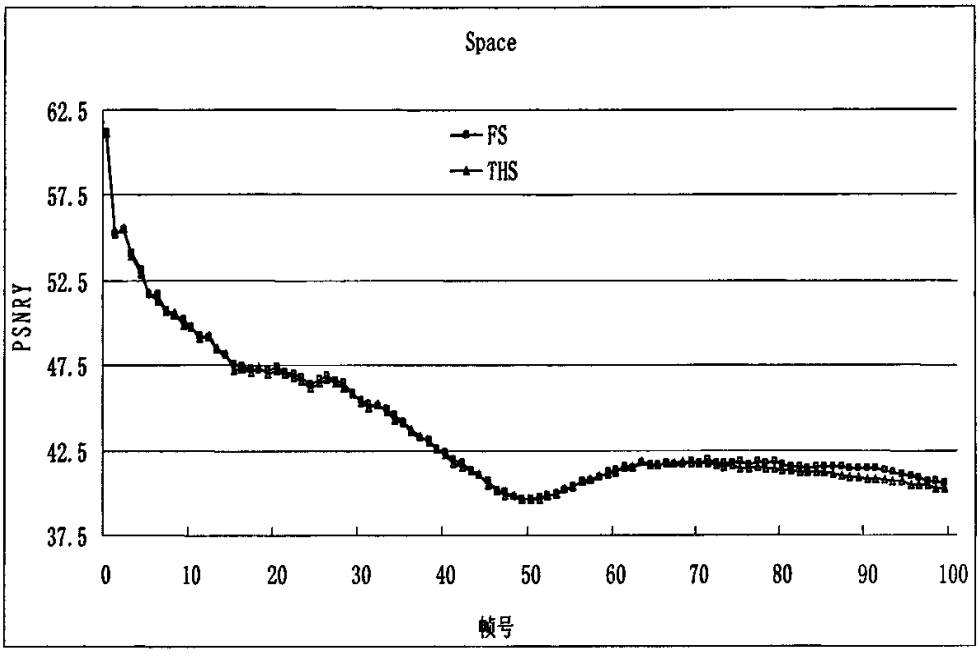

图 3.24 峰值信噪比比较

Circus 是 352×288 中等运动矢量图像序列。分别采用半像素 FS 和 THS 搜索算法进行处理，得到的性能参数表和峰值信噪比的曲线图如表 3-9、图 3.25 所示。

表 3-9 FS 和 THS 搜索算法的性能参数

性能 / 图像	FS 搜索次数/MB	THS 搜索次数/MB	THS比FS 运算减少量	FS PSNRY（dB）	THS PSNRY（dB）	THS比FS 减少（dB）
Circus	8.00000	1.97222	75.34725%	36.08229	35.97601	0.10628

针对此图像序列，THS 算法相对 FS 算法每帧平均减小 2385.79 次 SAD 计算，搜索次数减小 75.34725%。就 PSNR 而言，THS 比 FS 算法得到的 PSNR 减小 0.10628dB。

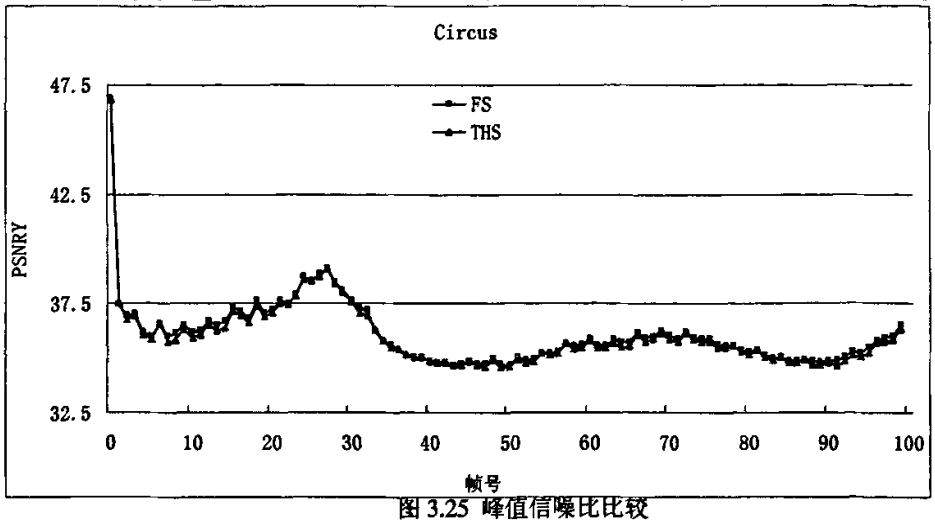

图 3.25 峰值信噪比比较

Man Wakling 是 176×144 大运动矢量图像序列图像序列。分别采用半像素 FS 和 THS 搜索算法进行处理，得到的性能参数表和峰值信噪比的曲线图如表 3-10、图 3.26 所示。

表 3-10 FS 和 THS 搜索算法的性能参数

性能 / 图像	FS 搜索次数/MB	THS 搜索次数/MB	THS比FS 运算减少量	FS PSNRY（dB）	THS PSNRY（dB）	THS比FS 减少（dB）
Man Walking	8.00000	1.90222	76.22225%	36.03584	35.84060	0.19524

针对此图像序列，THS 算法相对 FS 算法每帧平均减小 603.68022 次 SAD 计算，搜索次数减小 76.22225%。就 PSNR 而言，THS 比 FS 算法得到的 PSNR 减小 0.10628dB。

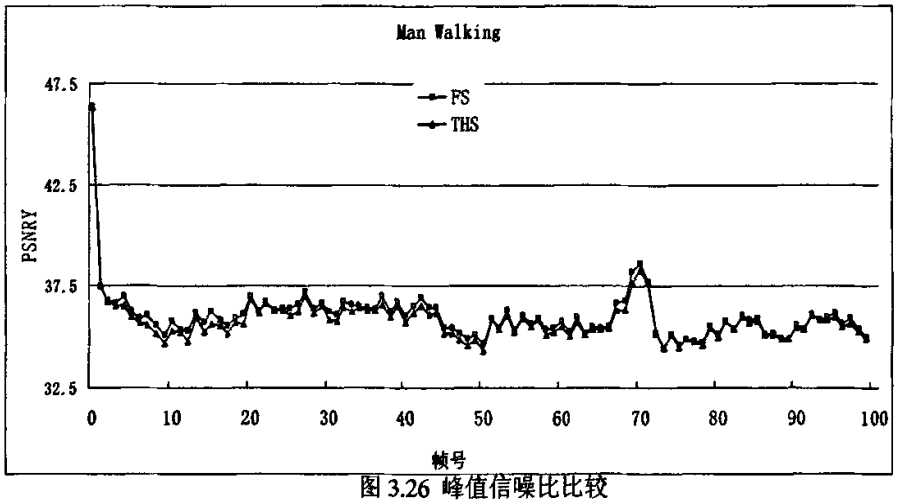

图 3.26 峰值信噪比比较

3.11 本章小结

本章首先全面地分析了 MPEG4 中多模式块匹配运动估计的编码框架；针对经典快速算法中的一些固有缺陷，在搜索速度和搜索精度这两个方面进行了折衷处理，利用中心偏置特性和优化的搜索策略提出三种新型快速算法——趋势菱形算法(TDS)和形状自适应算法，一种新型半像素快速算法——趋势半像素算法(THS)，运用于大、中、小运动矢量图像序列并取得各种性能参数与 FS 和 DS 经典算法进行对比，结果显示在获得与 FS 和 DS 算法相当的图像质量和信噪比的情况下，该算法有效地减少了搜索次数，提高了搜索效率，其中趋势菱形算法对中大运动矢量的视频图像运动估计效果更为明显，而形状自适应算法是目前已知的快速算法中是最快的一种算法，趋势半像素算法也将半像素的搜索速度极大的提高。从而证明了本文所提出的方法的正确性和高效性。

第四章 Video Analyzer 的设计与完善

4.1 Video Analyzer 的功能介绍

Video Analyzer 视频分析工具具有五大功能：VOP 和 MB 提示信息、视频导航细节视图、TRACE 跟踪功能、告警功能、BUFFER 校验模型。

如图 4.1 其中 VOP 提示信息包括：图像大小（如 352×288CIF 格式）、图像格式（在 MPEG4 中只有 YUV 4:2:0 这种图像格式）、码流大小、当前帧在在码流中的起始地址、当前帧的显示时间、当前帧类型（如 IVOP、PVOP、BVOP、SVOP）、当前帧的显示序号、当前帧的解码序号、编码当前帧头部所用比特数、编码当前帧宏块所用比特数、视频标准（如 MPEG4、简单档次/级别 2）。

如图 4.1 其中 MB 提示信息包括：被选中高亮宏块所在的 VOP 序号、被选中高亮宏块左上角像素位置、被选中高亮宏块在图像中的宏块位置、被选中高亮宏块在码流中的起始地址、被选中高亮宏块的类型（对于 IVOP 有 INTRA、INTRAQ、SKIP 这三种宏块类型；对于 PVOP 有 INTRA、INTRA+Q、INTTER、INTTER+Q、INTER4V、SKIP 这六种宏块类型；对于 BVOP 有 FORWARD、BACKWARD、INTERPOLATE、DIRECT 这四种宏块类型）、被选中高亮宏块的量化步长、编码被选中高亮宏块所用的比特数、被选中高亮宏块编码过程中是否有 AC 预测、被选中高亮宏块的编码块模式、被选中高亮宏块的运动矢量（IVOP 中的宏块无运动矢量；PVOP 中类型为 INTRA、INTRA+Q 的宏块无运动矢量，类型为 INTTER、INTTER+Q 的宏块各有一个前向运动矢量，类型为 INTER4V 宏块各有四个前向运动矢量；BVOP 中类型为 FORWARD 的宏块有一个前向运动矢量，类型为 BACKWARD 的宏块有一个后向运动矢量，类型为 INTERPOLATE 的宏块有一个前向和一个后向运动矢量，DIRECT 类型的宏块有四个前向和四个后向运动矢量）。

如图 4.2 视频导航细节视图：每一帧的显示序号、图像的类型、解码序号、显示时间、图像的比特数、图像的开始地址。

如图 4.3TRACE 跟踪功能:对于 MPEG1、MPEG2、MPEG4、H.264、VC1 的码流从开始码到结束码，所有比特都可以进行解释和说明。包括 parse bitstream 、interpret、VOP summary、MB row summary、MB summary、DCT level、pixel level 等功能。

如图 4.4 和图 4.5 告警功能：当进行解码时遇到不符合标准的地方，弹出告警窗口对错误进行描述。如错误的级别（致命错误、一般性错误、警告）、在那一帧什么位置、引起错误的原因。有两个选项：“Skip this Error only in future”，即可跳过以后此类错误信息继续解码。“Skip ALL Error alerts in future”，即可跳过以后各种非致命错误信息继续解码。

如图 4.6 BUFFER 校验模型：从 VBV 参数、或档次和级别指示符、或视频对象类型指示符中的一种或多种组合（优先级依次下降）获取初始化参数，如果码流中只有视频对象类型指示符，那么该选择该档次最高级别的始化参数。随着解码的进行获得数据并画出 VBV、VMV、VCV 的曲线图以此判断编码器编码出来的码流是否符合特定的档次和级别规定，有无 BUFFER 上溢或下溢。

如图 4.7 所示 DCT level trace 功能：VIDEO ANALYZER 不仅完成了 VPROVE 的所有功能，而且进行了功能升级，对隔行码流也能进行解码和分析，这是 VPROVE 分析工具所不具备的。虽然对于 MPEG4 隔行码流目前还不普及，目前流行的解码器包括 MPC、REALONE、暴风影音等都无法解码和播放 MPEG4 隔行码流，但是作为对国际标准的学习与研究，还是很有理论和应用价值。

图 4.8 image inspector 功能：对高亮显示的宏块每个亮度块进行分析，有传输的残差系数矩阵、变换前的系数矩阵、预测系数矩阵、最后的显示矩阵。

图 4.10 MV 示意图功能：将每个宏块的运动矢量直观的显示出来。

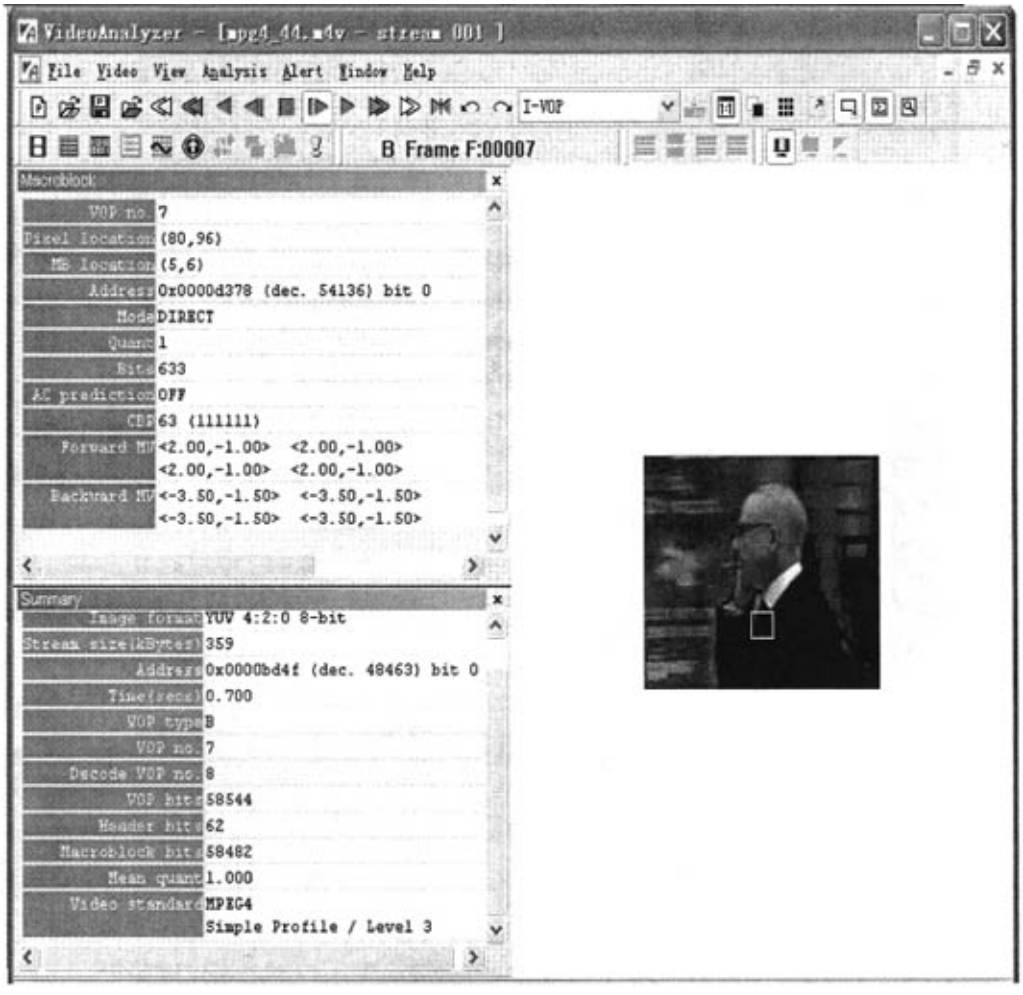

图 4.1 VOP 和 MB 提示信息

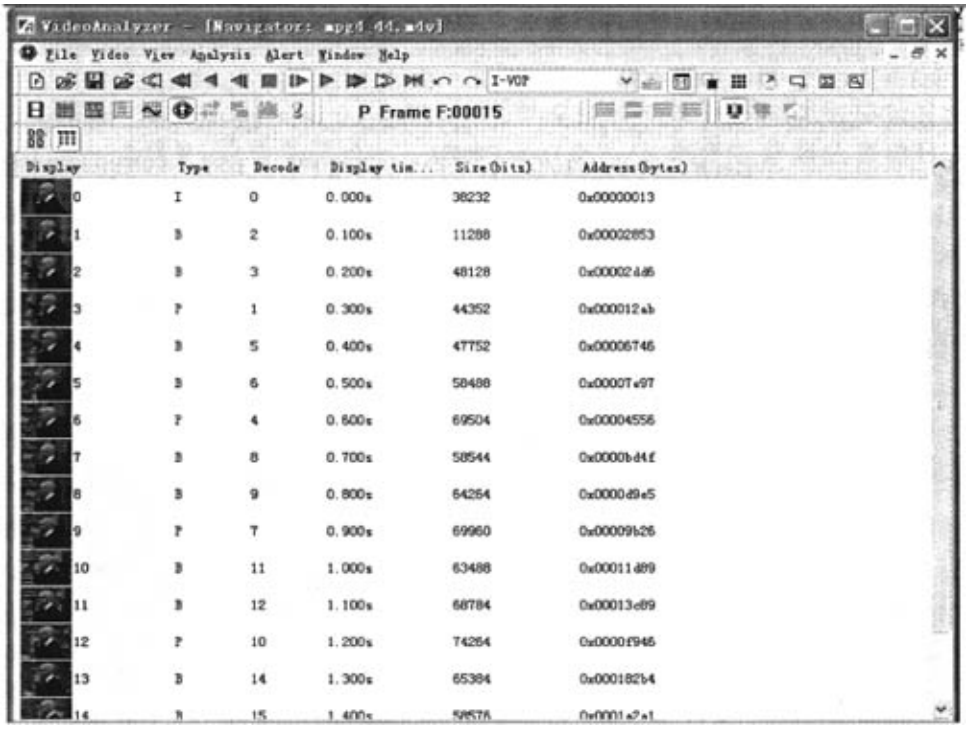

图 4.2 视频导航细节视图

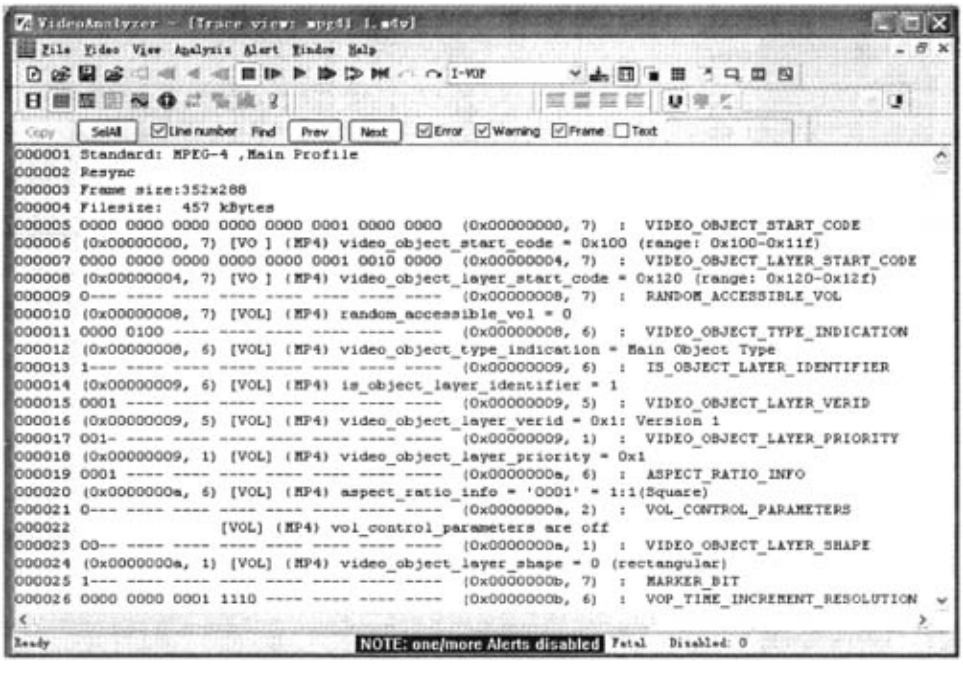

图 4.3 TRACE 跟踪功能视图

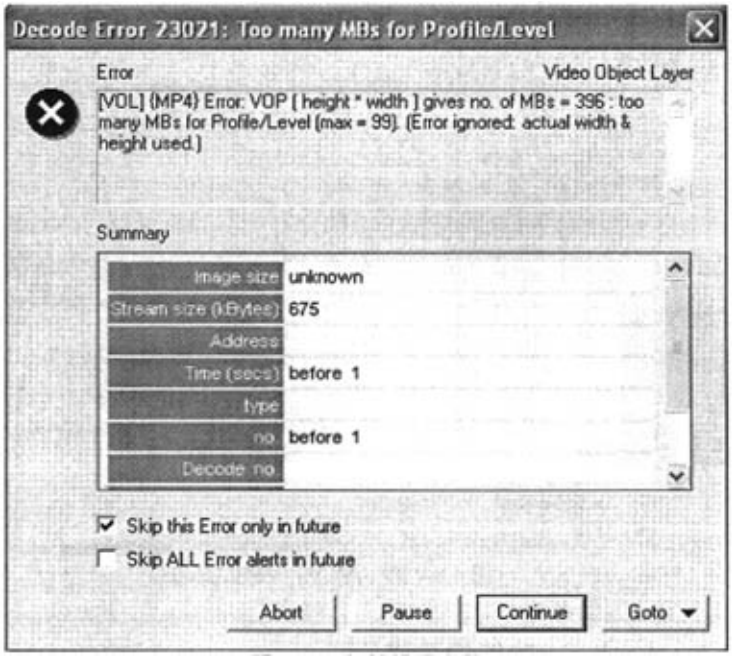

图 4.4 一般性错误告警

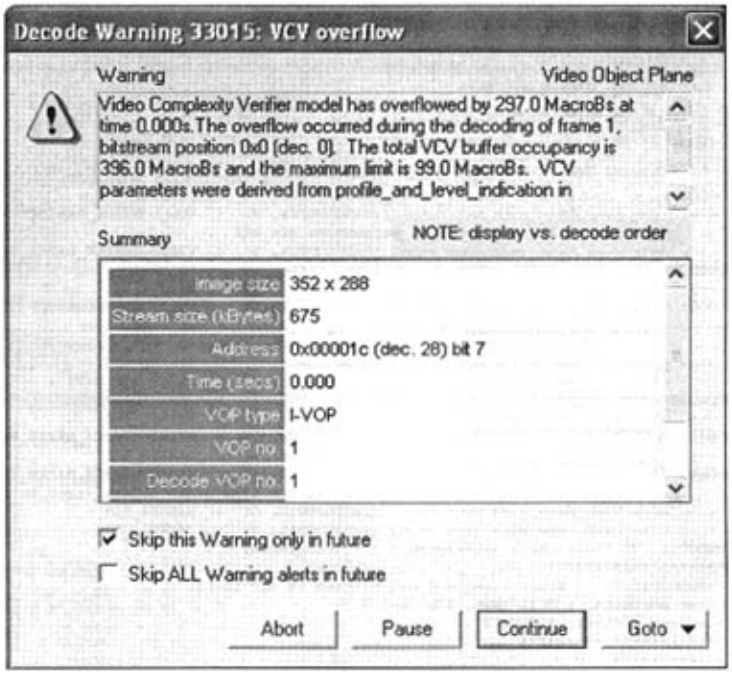

图 4.5 一般性警告

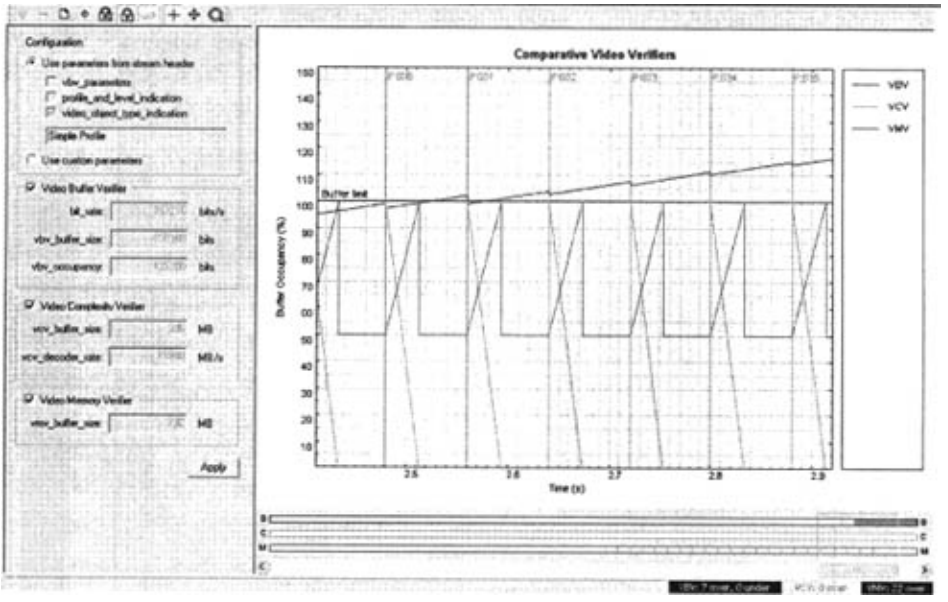

图 4.6 VBV、VCV、VMV

4.2 对 Video Analyzer 功能的改进

图 4.7DCT level 跟踪

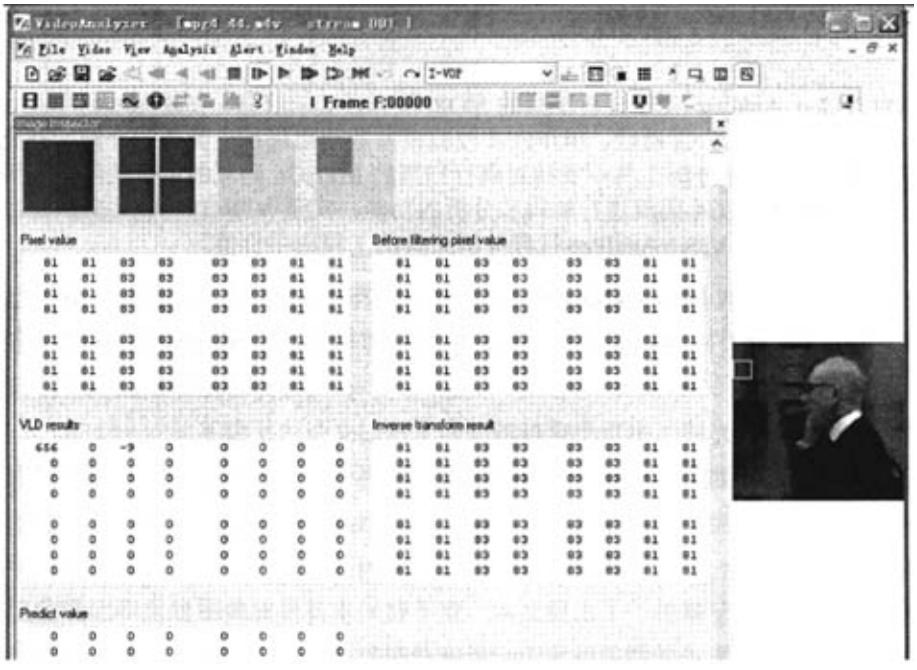

图 4.8 image inspector

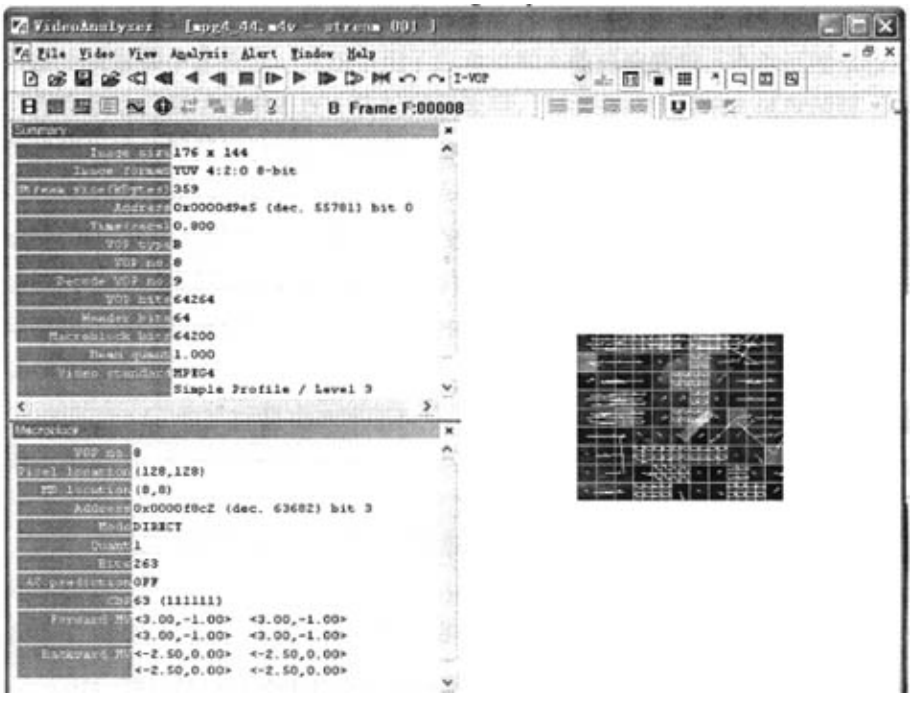

图 4.10 MV 示意图

4.3 本章小结

本章对 Video Analyze 视频分析工具的 VOP 和 MB 提示信息、视频导航细节视图、TRACE 跟踪功能、告警功能、BUFFER 校验模型等功能进行了简单的介绍。补充完善 VPROVE 视频分析工具不支持对隔行扫描的 MPEG4 码流的原有缺陷，实现对隔行扫描的 MPEG4 码流进行解码和分析的功能，实现 VPROVE 视频分析工具的功能升级，提升 Video Analyze 视频分析工具的工程应用价值。

第五章 结论

5.1 文研究的主要成果

论文首先分析研究了运动估计中块匹配快速算法，并针对提高算法效率时存在的初始搜索点选择、匹配准则和搜索策略三个主要问题，分别分析讨论和比较目前常用的算法的优缺点。

在上一点的基础上，论文提出了——趋势菱形算法、形状自适应算法、趋势半像素算法，并成功地应用于视频处理，该新算法在搜索速度、计算量、传输残差比特数、重建图像峰值信噪比等具体参数方面表现出较为优良的性能。其中形状自适应算法是目前所有时兴的快速算法中搜索速度最快的一种快速算法，而峰值信噪比和重建图像质量基本保持不变。本文提出了一种基于三角形的形状自适应搜索算法，从运动矢量场的连续性和运动矢量相关性的角度出发，根据视频序列的运动类型自适应的选择不同的搜索策略，同时对大运动块使用了起始点预测，对静止块直接中止搜索，算法采用了三角形和小菱形搜索法，在保证搜索精度的同时有效地提高了搜索速度。实验结果表明，形状自适应算法在速度超过了以往的快速运动估计算法以较小的代价，得到了与全搜索法相当的估计效果。

论文工作还包括设计 Video Analyze 视频分析工具，不仅完成了微软公司的 VPROVE 视频分析工具所有的功能，而且还进行了补充和完善，改进了该工具不支持对隔行扫描的MPEG4码流解码和分析的原有缺陷，实现对隔行扫描的MPEG4码流进行解码和分析的功能，实现 VPROVE 视频分析工具的功能升级，提升该工具的应用价值。

5.2 下一步需要开展的工作

本论文的研究还有许多地方有待进一步深入与扩展。作者认为有必要在以下方面继续研究：

1. 任意形状的块匹配自适应改算法。运动估计算法的匹配块不再局限于矩形，可以自适应改变其大小和形状，搜索路径趋于多样化这都是今后要进一步研究的问题。

参考文献

[1] 钟玉琢等. 基于对象的多媒体数据压缩编码标准——MPEG4 及其校验模型[M].北京：科学出版社，2000.

[2] ITU-T. Draft ITU-T recommendation: Video Coding for Low Bit Rate Communication. International Telecommunication Union, 26 September 1997.

[3] Koga T, Linuma K, Hirano A. Motion-compensated interframe coding for video conferencing. In: Proceedings of NTC81, 1981 ,961~965.

[4] Jain J R, Jain A K. Displacement measurement and its application in interframe image coding [J]. IEEE Transactions on Communications, 1981, COM-29(12):1799~1808.

[5] Ghanbari M. The cross-search algorithm for motion estimation. IEEE Trans Communication, 1990, 38(7) :950~953.

[6] Lee W , Wang J F , Lee J Y . Dynanic Search-window Adjustment and Interlaced Search for Block-matching Algorithm , IEEE Trans,CASVT,1993-0.

[7] Li R , Zeng B , Liou M L . A New Three-step Search Algorithm for Block Motion Estimation. IEEE Trans. CASVT,1994,4(4):438~442.

[8] Po M L, Ma W C. A Novel Four-step Search Algorithm for Fast Block Motion Estimation .IEEE Trans CASVT,1996,6(3):313~317.

[9] Liu L K, Feig E.A Block-based Gradient Descent Search Algorithm for Block Motion Estimation in Video Coding.IEEE Trans.CASVT, 1996,6(4):419~422.

[10] Luo L J, Zou C R, Gao X Q.A New Prediction Search Algorithm for Block Motion Estimation in Video Coding, IEEE Trans,on Consumer Electronics,1997,43(1):50~60.

[11] Xu J B, Po L M, Cheng C K. Adaptive Motion Tracking Block Matching Algorithms for Video Coding. IEEE Trans. On Circuits and System for Video Technology, 1999, 9(7): 1025~1029.

[12] Shan Zhu, Kai-Kuang Ma, A new diamond search algorithm for fast blockmatching motion estimation. IEEE Transaction on Information, Communications and Signal Processing. 1997,9 (2): 287~290.

[13] Yufei Yuan, Choong Wah Chan (Singapore). Overlapped Multi-resolution Motion Compensation Technique for Wavelet Video Compression. Visual Communicational and

Image Processing 2000, Proceedings of SPIE Vol.4067 (2000).

[14] 李涛, 余斯乐. 序列图像编码系统中基于小波变换的运动估计与补偿方法[J]. 中国图像图形学报, 1998,3(3).

[15] 沈兰荪，卓力，等.视频编码与低码率视频传输[M].北京:电子工业出版社，2001.

[16] Jianning Zhang, Yuwen He, et al. Performance and complexity joint optimization for H.264vieo coding [A]. Proceeding of international symposium on circuit system [C]. Bankok: PISCS, 2003.888~891.

[17] 刘海峰, 郭宝龙, 冯宗哲. 用于块匹配运动估计的正方形——菱形搜索算法. 计算机学报, 2002,25(7): 747~752.

[18] 吴红叶, 夏良正. 一种利用块间相关性的块运动估计算法[J]. 东南大学学报, 1997,27(2).

[19] 胡琳蓉, 朱秀昌.一种适合与 H.263 的运动估计搜索算法[J].通信学报,2000,26(6):65~69.

[20] Cheung C H, Po L M. A novel cross-diamond search algorithm for fast block motion estimation. IEEE Transactions on Circuits and Systems for Video Technology, 2002, 12(12): 1168~1177.

[21] Cheng F H, Sun S N. A New fast and efficient two-step search algorithm for block motion estimation. IEEE Transaction Circuits and Systems for Video Technology, 1999,9 (7): 977~983.

[22] Zhu C, Lin X, Chau L P. Hexagon-based search pattern for fast block motion estimation[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2002,12(5): 349~355

[23] Panusopome K, Baylon D M. An analysis and efficient implementation of half-pel motiom estimation. IEEE Transactions on Circuits and Systems for Video Technology, 2002, 12(8): 724~729.

[24] ISO/IEC JTC1/SC29/WG11 N3908. MPEG4 Video Verification Model Version 18.0. Jan.2001.

附录 英文缩写解释

CCITT	Consultative Committee on International Telegraphs and Telephones 国际电报电话咨询委员会	
ITU-T	International Telecommunications Union for Telegraphs and Telephones Sector 国际电信联盟电报电话通信部门	
JPEG	Joint Photographic Expert Group 联合组成的联合图像专家组	
BMME	Block Matching Motion Estimation	块匹配运动估计
MPEG	Moving Picture Expert Group	运动图像专家组
VO	Video Object	视频对象
VOL	Video Object Layer	视频对象层
VOP	Video Object Plane	视频对象平面
FDP	Facial Definition Parameter Set	人脸定义参数集
FAP	Facial Animation Parameter Set	人脸活动参数集
ME	Motion Estimation	运动估计
MC	Motion Compensation	运动补偿
FS	Full Search	全搜索算法
TSS	Three Step Search	三步搜索算法
NTSS	New Three Step Search	新三步搜索算法
TWSS	Two Step Search	两步搜索算法
CSA	Cross Search Algorithm	十字交叉算法
DS	Diamond Search	菱形搜索算法
TDS	Trend Diamond Search	趋势菱形搜索算法
THS	Trend Half-pixel Search	趋势半像素搜索算法

作者攻硕期间取得的研究成果

[1] 欧阳国胜，罗永伦．一种用于视频编码运动估计的新算法.通信与信息技术(增刊), pp391-394, 2006, 12