论 文

基于 DHS-NPDS 的低功耗运动估计硬件结构设计

陈运必^{①*}, 李正东^②, 郭立^①, 谢锦生^①, 赵龙^①

^① 中国科学技术大学电子科学与技术系, 合肥 230027

^② 重庆通信学院电子技术教研室, 重庆 400035

* 通信作者. E-mail: ahbi@mail.ustc.edu.cn

收稿日期: 2011-03-26; 接受日期: 2011-04-25

国家自然科学基金 (批准号: 61071173) 和中国科学技术大学研究生创新基金资助项目

摘要 为了达到实时视频编码的低功耗、低带宽、省资源 3 个要求, 文中基于改进的归一化部分失真搜索算法, 提出一种新颖的运动估计硬件结构. 新结构同时支持归一化部分失真搜索和自适应搜索区域调整. 前者可降低运动估计的计算复杂度, 从而实现低功耗省资源两个要求, 后者能避免不必要的外存访问, 从而降低数据带宽. 在 UMC 90 nm CMOS 工艺下实现结果表明, 相比于传统结构的最好结果, 文中结构以 6.2% 的吞吐率损失, 换取面积效率和功耗效率分别提高 425.5% 和 397.5%.

关键词 部分失真搜索 自适应搜索区域 低功耗 低带宽 流水线 大规模集成电路

1 引言

运动估计是去除视频时间冗余的主要方法, 而块匹配算法, 因其简单高效, 已成为视频编码标准的核心模块^[1]. 最直接的块匹配算法是全搜索, 但全搜索计算量大, 即使是已优化的系统, 运动估计仍然占系统 50%~90% 的计算量^[2]. 实时视频编码应用中 (如无线会议), 速度、功耗、编码效率是首要关注的指标^[3].

削减运动估计计算量的主要技术是快速搜索和快速匹配. 快速搜索按规定的搜索模式对少数几个候选块进行匹配给出次优解, 但是信噪比损失相对较高, 且算法的数据依赖性强, 搜索路径不规则, 不易于硬件实现^[4]. 快速匹配按规则的螺旋搜索路径, 使用部分失真消除策略 (partial distortion elimination, PDE) 来快速排除非最佳候选块, 相比于全搜索, 可节约 92%~99% 的计算量, 而平均信噪比损失仅为 0.08 dB^[3]. 计算量下降致使功耗下降, 快速匹配适合于实时低功耗应用, 但算法流程复杂, 硬件实现难度较大. 目前仅有文献 [5,6] 将简单的 PDE 整合到运动估计硬件结构中, 但仍未充分发挥 PDE 的优势, 且存在数据带宽过高的不足.

运动估计对外存的频繁访问造成功耗过大, 会制约编码器的性能, 解决这一问题的有效方法是自适应搜索区域调整. 搜索区域包括搜索中心和搜索范围两个参数, 均可由已知的相邻运动矢量或者中心候选块的绝对差值和 (sum of absolute differences, SAD) 计算得出. 自适应搜索区域调整计算量小且

引用格式: 陈运必, 李正东, 郭立, 等. 基于 DHS-NPDS 的低功耗运动估计硬件结构设计. 中国科学: 信息科学, 2012, 42: 527-536

对运动估计性能影响甚微, 相比于使用固定大小搜索区域, 支持自适应搜索区域的运动估计可节约 80% 的外存访问量, 而视频质量仅下降 0.02 dB^[7]. 自适应搜索区域调整的另一好处是缩小了平均的搜索范围, 需要匹配的候选块数目急剧减少, 可避免不必要的匹配计算.

本文关注快速匹配和自适应搜索区域调整两项技术在运动估计硬件设计中的运用, 以文献 [3] 中改进的归一化部分失真搜索 (dual-halfway-stop normalized partial distortion search, DHS-NPDS¹⁾) 为基础, 提出一种新颖的运动估计硬件结构, 同时达到低功耗、低带宽、省资源 3 个要求. 新结构的核心技术包括: 1) 支持采样点组读取的搜索区域缓存, 2) 可回退的部分失真计算流水线. 在 UMC 90 nm CMOS 工艺下实现结果表明, 相比于传统结构, 本文结构以不影响系统性能的少量吞吐率损失, 换取面积效率和功耗效率的显著改善.

2 设计相关的算法原理

DHS-NPDS 流程图如图 1 所示, 可分为两个子过程: 1) 计算第一个候选块 (即搜索区域的中心候选块) 的 SAD, 判断其是否是零矢量宏块, 若是则结束任务, 若不是则计算自适应搜索范围, 并判断搜索是否已经超出规定范围, 若是则结束任务, 若不是则继续其余候选块的匹配计算; 2) 对其余候选块按归一化部分失真消除 (normalized partial distortion elimination, NPDE) 方式进行迭代计算, 快速排除非最佳候选块, 每个候选块计算完毕都判断当前最小 SAD 是否小于终止阈值或搜索是否已超出规定范围, 满足以上两者之一则结束任务, 都不满足则继续下一候选块计算.

下面描述与设计相关的 DHS-NPDS 算法原理. 第 p 个部分失真 d_p 与第 p 个部分失真累加和 D_p 定义如下:

$$d_p = \sum_{i,j=0,4,8,12} |a_{i+s_p,j+t_p} - b_{i+s_p,j+t_p}|, \quad (1)$$

$$D_p = \sum_{k=1}^p d_k, \quad (2)$$

其中 (s_p, t_p) 为第 p 采样点组的偏移 (图 2), $a_{i+s_p,j+t_p}$ 和 $b_{i+s_p,j+t_p}$ 是分别来自宏块和候选块的采样点组数据. 算法的核心是通过累加计算 D_p , 其中 $p = 1, 2, \dots, 15$, 直到满足条件 $16D_p > p \cdot f \cdot D_{\min}$, 即可判定当前候选块为非最佳候选块, 从而跳过当前候选块的剩余计算. 判定条件中的松弛因子 f 可根据信噪比要求灵活取值, 默认取值为 $9/8$, $D_{\min}$ 为当前最佳候选块的 SAD. 当 $p = 16$ 时, 直接判断 $D_{16} < D_{\min}$, 若成立则当前候选块成为当前最佳候选块并修改 $D_{\min} = D_{16}$, 反之, 继续下一候选块的计算.

DHS-NPDS 同时使用终止阈值和自适应搜索区域调整两项技术, 有效地削减运动估计所需的数据量和计算量. 终止阈值 $T_{\text{DHS},q}$ 和自适应搜索范围 $\text{ASR}_{n,q}$ 可由第一个候选块的 SAD 计算得出, 这两个参数在整个宏块的运动估计过程中只需计算一次且涉及乘法和除法, 不适宜直接用硬件电路计算. 较经济的方案是, 运动估计引擎完成第一个候选块计算后请求 CPU 中断服务, 中断服务程序读取第一个候选块的 SAD 并计算 $T_{\text{DHS},q}$ 和 $\text{ASR}_{n,q}$, 再将其写入引擎的相应寄存器, 然后启动引擎进行其余候选块的计算, 如果不需要进行其余候选块计算, 则直接对引擎进行初始化, 为下一宏块的运动估计任务做准备.

1) DHS-NPDS 已被 joint video team (JVT) 采纳作为 H.264 参考软件的推荐运动估计算法

图 1 DHS-NPDS 流程图

Figure 1 Flowchart of DHS-NPDS

图 2 部分失真计算的采样点组及其序号

Figure 2 Computation order of partial distortion d_p of 16×16 block

为了支持自适应搜索区域数据存取, 借鉴文献 [7] 的存储器规划方案, 以 64×64 作为搜索区域的最大尺寸, 相应的最大搜索范围是 49×49 . 如图 3 所示, 相邻的两个宏块 i 和 j 的实际搜索区域分别为图中左右两个粗虚线方框, 宏块 j 的最大搜索区域为图中实线方框. 宏块 j 的实际搜索区域 (48×40) 比最大搜索区域 (64×64) 要小一半以上. 为了充分利用外存的块传输模式, 将整个搜索区域划分为 8×8 子块, 并将其作为外存读取的基本单位. 在对宏块进行运动估计之前, 将落在或部分落在实际搜索区域内且尚未写入的子块从外存写入片内搜索区域缓存. 图 3 中同时标注 i 和 j 的子块为宏块 i 和宏块 j 搜索区域的重叠数据, 从宏块 i 切换到宏块 j , 只需写入仅标注 j 的 14 个新子块. 为了实现相邻搜索区域的数据重用 [8], 互联搜索区域缓存的左右边沿构成循环缓存. 因为水平方向宏

图 3 自适应搜索区域及其存储规划

Figure 3 Layout of the search window cache

块切换时, 搜索区域整体向前移位 16 个像素, 所以 64×64 搜索区域有 4 种工作模式: 1) $0 \rightarrow 63$, 2) $16 \rightarrow 63 \rightarrow 0 \rightarrow 15$, 3) $32 \rightarrow 63 \rightarrow 0 \rightarrow 31$, 4) $48 \rightarrow 63 \rightarrow 0 \rightarrow 47$.

3 DHS-NPDS 运动估计引擎的优化设计

图 1 的算法流程图表明, DHS-NPDS 具有很强的串行性, 按螺旋搜索路径, 候选块只能逐个进行计算, 不易并行处理, 另外, 同一候选块内的部分失真迭代也只能串行执行. 以上两个因素会制约运动估计引擎的数据吞吐率, 为了提高硬件实现效率, 需另寻其他实现途径. 幸运的是, 统计表明, 即使是包含激烈运动的视频 (如 FOOTBALL²⁾), 只需进行一次部分失真迭代的候选块数目比重也超过 95%, 该结论为构造高效的部分失真计算流水线提供了保障. 运动估计引擎提供两个工作模式, 候选块间迭代和候选块内迭代. 默认情况下, 引擎处于候选块间迭代模式, 即按一个候选块进行一次部分失真迭代的方式流水运行. 出现需要多次部分失真迭代的候选块时, 打断流水线并保存中间数据, 然后切换到候选块内迭代模式, 直到完成该候选块的计算, 再切换回候选块间迭代模式. 本文将这种特性的结构称为可回退流水线结构.

3.1 硬件架构

DHS-NPDS 硬件架构如图 4(a) 所示, 首先 CPU 将第一个候选块的数据和宏块的数据分别写入搜索区域缓存和宏块寄存器阵列, 然后启动运动估计引擎. 引擎先对第一个候选块进行 16 次部分失真累加, 所得 SAD 作为 $D_{\min}$ 的初始值, 再向 CPU 发出中断请求. 中断服务程序读取 $D_{\min}$ 并计算 $T_{\text{DHS},q}$ 和 $\text{ASR}_{n,q}$, 再将其写入引擎的相应寄存器, 如果搜索范围内的数据还未完全写入搜索区域缓存, 则将所需数据从外存读出并写入搜索区域缓存, 最后再启动引擎进行后续搜索. 引擎按螺旋搜索路径对其他候选块进行匹配计算, 直至满足终止阈值条件或者搜索范围溢出, 才第二次向 CPU 发出中断请求. CPU 读取运动估计结果, 并对引擎进行复位, 至此, 当前宏块的匹配任务完成. 计时器在每次匹配任务开始时初始化, 每当出现更小的 SAD, 时间寄存器 ($D_{\min}$ win-time) 记录胜出时刻. 因为搜索路径是已知的, 计时器时刻和搜索点一一对应, 记录胜出时刻等价于记录当前最佳候选块对应的运动矢量 [9].

2) Xiph.org test media. <http://media.xiph.org/video/derf/>

图 4 硬件架构与搜索区域地址生成机构

Figure 4 System architecture of HDS-NPDS and address generator for partial distortion iteration. (a) Iterative loop of partial distortion; (b) address generator for partial distortion iteration

上述记录方式简化了控制逻辑的设计且不需额外的运动矢量保存电路, 既节约面积又降低功耗.

图 4(a) 中加粗的组件和路径构成部分失真迭代环路, 对环路不能直接插入寄存器以形成多级流水线^[10]. 本文采用可回退的 2 级流水线结构, 其关键在于图 4(b) 所示的地址生成机构. 引擎处于候选块间迭代模式, 则以块间地址作为搜索区域缓存的读地址, 同时旧地址寄存器存储着上一个候选块地址, 当切换到候选块内迭代模式, 可从旧地址寄存器调出上一候选块地址并对其进行块内的采样寻址. 对于更多级流水线结构, 地址生成单元将更为复杂, 需要记忆更多过往候选块的地址, 采用 2 级流水线结构可得到比较好的速度和硬件复杂度折中.

3.2 搜索区域缓存模块设计

搜索区域缓存需支持采样点组的读取和 8×8 子块的写入, 前者是进行部分失真迭代的基础, 后者是为了支持自适应搜索区域. 采样点组是 4×4 的像素矩阵, 组内两个水平或垂直相邻的像素点物理间隔 3 个像素. 采样点组用其左上角像素的地址进行标识, 即采样点组 (i, j) 对应的像素集合为 $\{(i + \varepsilon_m, j + \varepsilon_n) | \varepsilon_m, \varepsilon_n = 0, 4, 8, 12\}$. 为了同时读出整个采样点组, 需使得采样点组的 16 个像素点存储在不同的 SRAM 中. 如图 5 所示, 将 64×64 小的搜索区域划分为 $16 \times 16 = 256$ 个互不重叠的 4×4 子块, 子块 (u, v) 存储在 $\text{SRAM}(u\%4, v\%4)$ 中, 比如图 5 中所有灰色的 4×4 子块按从左到右自上而下的方式存储在 $\text{SRAM}(0, 0)$ 中, 同时 4×4 子块内的像素亦按从左到右自上而下的方式存储. 搜索区域缓存是 4×4 的 SRAM 存储阵列.

按图 5 的存储映射关系, 采样点组内任一像素点 $(i + \varepsilon_m, j + \varepsilon_n)$ 存储在 $\text{SRAM}(\lfloor (i + \varepsilon_m)/4 \rfloor\%4, \lfloor (j + \varepsilon_n)/4 \rfloor\%4)$ 中偏移地址为 $16 \times (4 \times \lfloor (i + \varepsilon_m)/16 \rfloor + \lfloor (j + \varepsilon_n)/16 \rfloor) + 4 \times ((i + \varepsilon_m)\%4) + (j + \varepsilon_n)\%4$ 之处. 将 4×4 地址矩阵关联到 SRAM 阵列前, 需对其进行适当的移位对齐. SRAM 阵列的规划如图 5 所示, 左上角为 $\text{SRAM}(0, 0)$, 未对齐的地址矩阵的左上角地址 (i, j) 应该关联到 $\text{SRAM}(\lfloor i/4 \rfloor\%4, \lfloor j/4 \rfloor\%4)$, 因此将地址矩阵循环下移 $\lfloor i/4 \rfloor\%4$ 次, 循环右移 $\lfloor j/4 \rfloor\%4$ 次, 以实现地址对齐. 为简化电路设计, 地址矩阵的 2 维移位可用 ε_m 和 ε_n 的 1 维移位实现. 最后, 从 SRAM 阵列中读出的 4×4 数据矩阵要进行逆向移位, 即循环上移 $\lfloor i/4 \rfloor\%4$ 次, 循环左移 $\lfloor j/4 \rfloor\%4$ 次, 才能对齐到自然次序, 也可以不改变 SRAM 阵列的读出数据的次序, 让宏块采样点组反面对齐到 SRAM 阵列的读出数据, 即宏块采样点组循环下移 $\lfloor i/4 \rfloor\%4$ 次, 循环右移 $\lfloor j/4 \rfloor\%4$ 次.

图 5 搜索区域缓存的存储映射关系图

Figure 5 Memory map of the search window cache. An 16×16 SRAM array is used to construct the cache

综合分析搜索区域缓存的存储规划 (图 2) 及其存储映射关系 (图 5) 可知, 为了实现向搜索区域缓存写入 8×8 数据块, 需将水平相邻的 SRAM 两两绑定为一组, 如图 5 的圆角方格所示, 共有 8 组, 组内两个 SRAM 共用写使能, 不同组使用不同写使能. 使用 64 bit 数据总线, 一个周期可向搜索区域缓存写入 8×8 数据块的一行数据. 以上写入方式要求所用 SRAM 的写入端口为 32 bit, 读出端口为 8 bit, 且写入地址 (i, j) 中的列地址 j 是 8 的整数倍. 读地址和写地址共享地址译码逻辑, 读写两种模式的区别仅在于, 写模式需要根据写入地址生成各组 SRAM 的写使能.

地址译码的数学过程看似复杂, 但是译码过程中的乘法、除法和取模运算的因子均为 2 的整数次幂, 映射到硬件电路就是位截取和位拼接操作, 实现起来非常简单. 优化后的地址译码电路如图 6 所示, 图 6(a) 为高位地址矩阵的生成电路, 包括循环缓冲模式控制电路 (circular 为循环模式控制信号) 和 ε_m 与 ε_n 的 1 维移位电路; 图 6(b) 为最终送入 $\text{SRAM}(m, n)$ 的物理地址. 从图 6 可知, 译码电路仅需要移位电路、加法器和一些附加逻辑, 位截取和位拼接操作不消耗逻辑资源.

3.3 部分失真计算与比较模块设计

部分失真 d_p 是宏块和候选块第 p 采样点组数据的绝对差值和, 假设 a_{ij}^p 和 b_{ij}^p 是分别来自宏块和候选块的采样点组数据, 其中 $i, j = 0, 1, 2, 3$. 为了缩短关键路径, 必须采用快速算法来计算差的绝对值, 见图 7(a), 首先计算差值 $a_{ij}^p - b_{ij}^p$, 如果借位信号 c_{ij}^p 为 0, 说明 $a_{ij}^p \geq b_{ij}^p$, 则 $|a_{ij}^p - b_{ij}^p| = a_{ij}^p - b_{ij}^p$, 如果借位信号 c_{ij}^p 为 1, 差值按位取反后再加 1 可得到 $|a_{ij}^p - b_{ij}^p|$. 上述计算过程如下式,

$$d_p = \sum_{i,j=0,\dots,3} |a_{ij}^p - b_{ij}^p| = \sum_{i,j=0,\dots,3} (c_{ij}^p + r_{ij}^p). \quad (3)$$

从图 4(a) 的硬件架构可知, c_{ij}^p 和 r_{ij}^p 被流水线寄存器锁存之后才作为下一级求和的输入.

图 7(b) 为部分失真计算与比较电路, 其求和输入来自流水线寄存器锁存的 c_{ij}^p 和 r_{ij}^p , 共 32 个数据, 其中 c_{ij}^p 为 1 bit, r_{ij}^p 为 8 bit, 使用进位保留加法 (carry save adder, CSA) 电路可以获得很好的面积和速度折中. $\text{REG}[D_{\min}]$ 用于保存当前最佳候选块的 SAD, 每开始新的候选块匹配计算, 就对 $\text{REG}[D_p]$ 和 $\text{REG}[p \cdot D_{\min}/16]$ 进行清零, 随后的候选块内迭代使用 $\text{REG}[D_p] + d_p \geq (\text{REG}[D_{\min}]/16 + \text{REG}[p \cdot D_{\min}/16]) \times 9/8$ 作为排除条件, $\text{REG}[p \cdot D_{\min}/16]$ 的内容会随着块内迭代不断更新. 比较器的结果直接传递给控制逻辑, 用于控制流水线的运行模式. 另外还可以设置终止阈值 $T_{\text{DHS},q}$, 用于终止

图 6 搜索缓存地址译码

Figure 6 Address decoding circuit of the search window cache. The node labeled “con” is operation of bit concatenation. (a) High-bit-segment address generator; (b) physical address connecting to SRAM(m, n)

图 7 做差单元以及部分失真计算与比较电路

Figure 7 Fast structure for absolute of difference and partial distortion summation-comparison unit. (a) Absolute of difference unit; (b) partial distortion summation-comparison unit

零运动矢量块的不必要计算, $T_{DHS,q}$ 可以是固定值 256 或者 512, 也可以使用预测公式给出的动态阈值^[3]. 图 7(b) 中的乘于 1/16 的乘法实为移位, 不占资源, 乘于 9/8 的乘法可用加法和移位实现, 即 $(x \gg 3) + x$, x 为另一个乘数.

4 实现结果分析

设计采用 Verilog-HDL 语言进行描述, 在 UMC 90 nm CMOS 工艺下使用 Synopsys 的 EDA 工具进行综合实现, 在 250 MHz 的时钟频率约束下, 消耗的等价门电路个数为 30 K, SRAM 容量为 4 KB, 全速工作时动态功耗为 84.4 mW. 由于使用自适应搜索区域, 运动估计引擎的处理帧率会随不同视频有所波动, 为了反映最坏情况下引擎的处理能力, 对包含激烈运动的视频序列 FOOTBALL 进行测试, 结果表明, 每个候选块的平均计算时间为 1.2778 个周期. 如果使用固定大小的 64×64 搜索区域, 也就是 49×49 的搜索范围, 引擎的处理能力是 $1280 \times 720 @ 22\text{fps}$, 处理一个宏块需要 $49 \times 49 \times 1.2778 \approx 3068$ 个周期. 实际运行时, 由于自适应搜索区域调整, 平均搜索范围要小一些, 另外也可通过限制最大搜索

表 1 与典型运动估计结构的实现结果比较

Table 1 Comparison with other ME engines for H.264/AVC IME

	Ref.[12]	Ref.[13]	Ref.[14]	Ref.[15]	Ref.[11]	the proposed
Process (μm)	0.13	0.18	0.18	0.18	0.18	90
Search range	32×32	16×16	65×65	65×33	33×33	49×49 (adaptive)
Gate count (K)	61	597	210	168	160	30
Frequency (MHz)	294	200	260	216	200	250
Power (mW)	570	503	—	—	423	84.4
TP	18 375 000	200 000 000	205 335 000	208 494 000	192 914 083	195 648 771(−6.2%)
E	301.2	335	977.8	1241	1205.7	6521.6(+425.5%)
E_p	32 236.8	397 614.3	—	—	465 967.1	2 318 113.4(+397.5%)

范围来得到较高的处理帧率, 仿真测试表明, 本文设计能胜任 $1280 \times 720@30\text{fps}$ 的运动估计任务.

为了定量评估本文结构的有效性, 定义 3 个参数^[11]: 数据吞吐率 TP、面积效率因子 E 和功耗效率因子 E_p , 如 (4)~(6) 式所示,

$$\text{TP} = \frac{f}{C} \times S_w \times S_h, \quad (4)$$

$$E = \text{TP}/G, \quad (5)$$

$$E_p = \text{TP}/P, \quad (6)$$

其中 f 为时钟频率, C 为完成一个宏块运动估计所需的周期数, $S_w \times S_h$ 为搜索范围, G 为设计的等效门个数, P 为动态功耗.

表 1 给出本文结构与以往结构的实现结果比较, 相比以往的最好结果, 本文结构以 6.2% 的吞吐率损失, 换取面积效率提高 425.5%, 功耗效率提高 397.5%. 面积效率的改善是因为本文结构消耗的等效门个数远小于以往结构. 基于 NPDE 的计算模块只需进行 16 个像素点的部分失真计算, 以往结构需要进行 $16 \times 16 = 256$ 个像素点的匹配计算, 仅是匹配计算电路就能节省不少面积. 功耗效率改善的原因在于, 超过 95% 的候选块只需进行一次迭代即被排除, 也就是每个候选块平均只需 1.2778 次部分失真计算, 以往结构每个候选块需要 16 次等价的部分失真计算, 本文结构的计算复杂度远小于以往结构的计算复杂度. 除了效率指标有明显改善, 本文结构的一大特点是支持自适应搜索区域调整, 因此运动估计可以灵活使用合适的搜索范围, 而不是固定的最大搜索范围 49×49 , 既可节约外存带宽又能降低运动估计的计算量, 从而节约不必要的功耗和计算时间, 稍微牺牲数据吞吐率不会造成系统瓶颈.

5 结论

本文提出一种新颖的基于 DHS-NPDS 的运动估计硬件结构, 同时支持归一化部分失真搜索和自适应搜索区域调整. 前者可显著改善硬件实现的面积效率和功耗效率, 后者能降低外存带宽要求. 针对新结构的两个核心模块——支持采样点组读取的搜索区域缓存和可回退的部分失真计算流水线, 给出了优化的设计方案. 在 UMC 90 nm CMOS 工艺下实现结果表明, 与传统结构的最好结果相比, 本文结构结构以 6.2% 的吞吐率损失, 换取面积效率和功耗效率分别提高 425.5% 和 397.5%. 以上实验数据表明, 本文结构比以往结构更能胜任苛刻的便携式视频编码要求.

参考文献

- 1 Richardson I E. H.264 and MPEG-4 Video Compression - Video Coding for Next-generation Multimedia. Chichester: Wiley, 2003. 30–41
- 2 Chen T C, Chien S Y, Huang Y W, et al. Analysis and architecture design of an HDTV720p 30 frames/s H.264/AVC encoder. *IEEE Trans Circ Syst Vid*, 2006, 16: 673–688
- 3 Yi X Q, Ling N. Improved normalized partial distortion search with dual-halfway-stop for rapid block motion estimation. *IEEE Trans Multimed*, 2007, 9: 995–1003
- 4 Chen Z B, Zhou P, He Y. Fast integer and fractional pel motion estimation for JVT. In: JVT-F017r, 6th Meeting of Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG. Awaji Island, 2002. 5–13
- 5 Do V L, Yun K Y. A low-power VLSI architecture for full-search block-matching motion estimation. *IEEE Trans Circ Syst Vid*, 1998, 8: 393–398
- 6 Jiang M, Crookes D, Davidson S, et al. Low-power systolic array processor architecture for FSBM video motion estimation. *Electron Lett*, 2006, 42: 1146–1147
- 7 Jung J, Kim J, Kyung C M. A dynamic search range algorithm for stabilized reduction of memory traffic in video encoder. *IEEE Trans Circ Syst Vid*, 2010, 20: 1041–1046
- 8 Tuan J C, Chang T S, Jen C W. On the data reuse and memory bandwidth analysis for full-search block-matching VLSI architecture. *IEEE Trans Circ Syst Vid*, 2002, 12: 61–72
- 9 Chen Y B, Guo L, Li Z D, et al. An efficient parallel architecture for one-bit transform based motion estimation. *J Electron Inform Technol*, 2011, 33: 717–722
- 10 Parhi K K. *VLSI Digital Signal Processing Systems: Design and Implementation*. New York: Wiley, 1999. 64–68
- 11 Cao W, Hou H, Tong J R. A high-performance reconfigurable VLSI architecture for VBSME in H.264. *IEEE Trans Consum Electr*, 2008, 54: 1338–1345
- 12 Yap S Y, McCanny J V. A VLSI architecture for variable block size video motion estimation. *IEEE Trans Circuits II*, 2004, 51: 384–389
- 13 Ou C M, Le C F, Hwang W J. An efficient VLSI architecture for H.264 variable block size motion estimation. *IEEE Trans Consum Electr*, 2005, 51: 1291–1299
- 14 Deng L, Gao W, Hu M Z, et al. An efficient hardware implementation for motion estimation of AVC standard. *IEEE Trans Consum Electr*, 2005, 51: 1360–1366
- 15 Li D X, Zheng W, Zhang M. Architecture design for H.264/AVC integer motion estimation with minimum memory bandwidth. *IEEE Trans Consum Electr*, 2007, 53: 1053–1060

Architecture design of low-power motion estimation based on DHS-NPDS for H.264/AVC

CHEN YunBi^{1*}, LI ZhengDong², GUO Li¹, XIE JinSheng¹ & ZHAO Long¹

¹ *Department of Electronic Science and Technology, University of Science and Technology of China, Hefei 230027, China;*

² *Staff room of Electronic and Technology, Chongqing Communication College, Chongqing 400035, China*

*E-mail: ahbi@mail.ustc.edu.cn

Abstract A novel architecture of motion estimation (ME) based on improved normalized partial distortion search is proposed to meet three primary requirements for real-time video encoding, which are low-power, low-bandwidth and high area utilization efficiency. The ME engine supports both normalized partial distortion search and adaptive search window adjustment. The former can reduce the computational complexity of ME to save

power and area; the latter can avoid unnecessary accessing the external memory to lower the data bandwidth. The proposed engine has been implemented with UMC 90nm CMOS technology. The implementation results show that, compared with traditional engines, the engine can achieve significant improvements of the hardware efficiency and the power efficiency respectively with a little throughput compromise.

Keywords partial distortion search, adaptive search range, low-power, low-bandwidth, pipeline, VLSI

CHEN YunBi was born in 1979. He received the Ph.D. degree in circuit and system from University of Science and Technology of China, Hefei, in 2010. Currently, he is an Electronic Engineer at Huawei Shanghai Research Institute. His research interests include low power design, digital signal processing, and system-on-chip designs.

LI ZhengDong was born in 1978. He received the M.S. degree in Circuit and System from University of Science and Technology of China, Hefei, in 2007. Currently, he is a Lecturer at Chongqing Communication College. His research interests include electronic system design and sensor networks.

GUO Li was born in 1946. Currently, he is a Professor and doctoral supervisor in the Department of Electronic Science and Technology at University of Science and Technology of China, Hefei. His research interests include video information analysis and processing.

XIE JinSheng was born in 1981. Currently, he is a Ph.D. student in the Department of Electronic Science and Technology at University of Science and Technology of China, Hefei. His research interests include signal processing, computer vision, action recognition and video analysis.