一种与编码器相结合的数字水印添加方式

黄磊磊 1)

1)(复旦大学 微电子学院, 上海 201203)

摘 要 数字水印技术，是一种基于内容的、非密码机制的计算机信息隐藏技术。该技术会将一些标识信息，即数字水印，直接或者间接地嵌入数字载体当中，但不影响原载体的使用价值，也不容易被探知和再次修改，却可以被生产方识别和辨认。先进主流的数字水印技术一共分成两大类，一类通过改变某些特定像素或量化系数的方式直接地嵌入水印信息；另一类则通过添加噪声的方式间接地嵌入水印信息。然而，上述方法都未结合编码器进行实施，因此，尽管水印引入的噪声可能在视觉感知上不易察觉，但却很有可能导致编码码率的大幅上升。基于这一原因，Mareen[1] 在《A Novel Video Watermarking Approach Based on Implicit Distortions》一文中提出了一种基于编码器的水印添加方式，而本文则在此基础上进行了若干优化。

关键词 数字水印，数字视频，编码器
中图法分类号 ????

An Encoder-Combined Embedding Method for Digital Watermarking

Huang Leilei 1)

1)(Department of Microelectronic, Fudan University, Shanghai 201203)

Abstract Digital watermarking is a kind of computer information hiding technology which is based on content instead of cipher mechanism. This kind of technology will embed some identification information, i.e. digital watermark, directly or indirectly into the digital carrier, but it will not affect the use value of the original carrier, and will not be easy to detect or modified. Advanced digital watermarking techniques are divided into two categories: one is embedding watermarking information directly by changing some specific pixels or quantization coefficients; the other is embedding watermark information indirectly by adding noise. However, none of the above methods are implemented in combination with the encoder. Therefore, although the noise introduced by the watermark may not be easily detected in visual perception, it is likely to cause a significant increase in coding rate. For this reason, Mareen[1] proposed a watermark adding method based on encoder in a new video watermarking approach based on implicit discards, and on this basis, this paper made some optimization.

Key words digital watermarking, digital video, encoder

1 引言

数字水印技术，是一种基于内容的、非密码机制的计算机信息隐藏技术。该技术会将一些标识信息，即数字水印，直接或者间接地嵌入数字载体当中，但不影响原载体的使用价值，也不容易被探知和再次修改，却可以被生产方识别和辨认。通过这些隐藏在载体中的信息，生产方可以达到确认内容创建者、购买者、传送隐秘信息或者判断载体是否被篡改等目的。数字水印是保护信息安全、实现防伪溯源、版权保护的有效办法，是信息隐藏技术研

究领域的重要分支和研究方向。本问所讨论的载体是视频文件。

现有的数字水印技术一般可以分为两大类。一类通过改变某些特定像素或量化系数的方式直接地嵌入水印信息。非常多的数字水印技术都选择将数据的最低比特作为改变的对象。其原因是：替换这些比特，既能够轻易地嵌入水印信息，又能够不轻易地被使用者所感知。然而，显然地，攻击者也能够通过改变这些比特来达到抹除水印信息的目的，并且，基于同样的原因，这些改变也不会显著地降低视频的质量。鉴于这一问题，一些算法提出了其他的修改方式。例如，Ga 等研究者[2] 提出对连续帧内帧的某些特定块的非零系数数量进行修改。Dutta 等研究者[3] 提出对连续帧内帧的某些特定块的前两个非零系数进行修改。

另一类则通过添加噪声的方式间接地嵌入水印信息。例如，Kalker 等研究者[4] 提出了基于频谱展开的噪声添加方式。Hartung 和 Yamada 等研究者[5, 6] 提出了基于伪随机种子的噪声添加方式。由于这些水印信息是通过噪声嵌入的，因此，对于这类水印，需要通过相关性检测的方式进行提取。确切地说，当一个被攻击（篡改）的视频与某个带水印的原始视频在相关性上达到一定程度时，即可认为该被攻击（篡改）视频的来源是这个带水印的原始视频。然而，由于不同的攻击手段会在不同程度上影响到视频的噪声情况，因此，很难给出一个统一通用的阈值。

通过上述分析可以看出，现有的技术在以下方面仍然存在一定的不足之处。

（1） 基于最低比特的技术在鲁棒性上较为不足，容易受到攻击。

（2） 基于噪音叠加的技术，尽管其在鲁棒性上有所提高，但在水印检测过程中所采用的阈值却难以统一。

（3） 上述方法都未结合编码器进行实施，因此，尽管水印引入的噪声可能在视觉感知上不易察觉，但却很有可能导致编码码率的大幅上升。

2 相关工作与预备知识

基于上述原因，《A Novel Video Watermarking Approach Based on Implicit Distortions》一文的作者 Hannes 提出了一种与编码器相结合的水印叠加方式。其核心原理是：

（1） 在对视频进行编码的过程中，我们可以通过改变某一个帧内块的预测模式来引入一定程度的失真。

（2） 一方面，由于编码器在数据上的高度依赖性，因此，这一失真大概率会被有效地传递。

（3） 另一方面，由于并未改变其他块的决策，因此，视频的失真仍然能够被控制在一定的范围内。

以下依次对上述原理进行展开：

2.1 失真的引入

以 HEVC 标准的帧内预测为例，每个帧内块都可以选择 35 种预测模式进行预测，包括 DC 模式，Planar 模式和 33 种角度模式。在绝大多数情况下，每种方式所得到的预测值都是不一样的。因此，改变预测方向就能够有效地改变预测像素，而改变预测像素会改变残差，改变残差又会改变由于量化所丢失的细节，并最终改变重建像素，从而引入失真。

下图分别展示了全部基于 Planar 模式，全部基于 DC 模式和全部基于第 18 种角度模式对序列 BlowingBubbles 进行预测所得到的预测图像，其差异性是非常显著的。

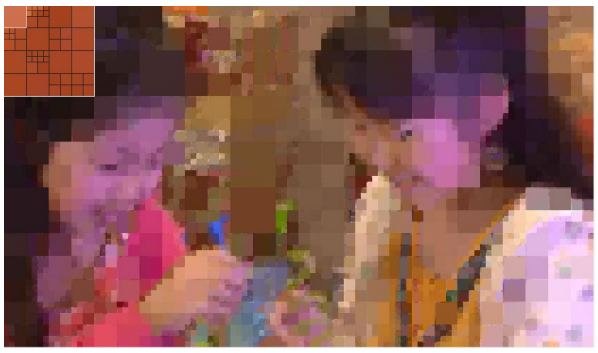

图 1 全部使用 Planar 模式进行预测

图 2 全部使用 DC 模式进行预测

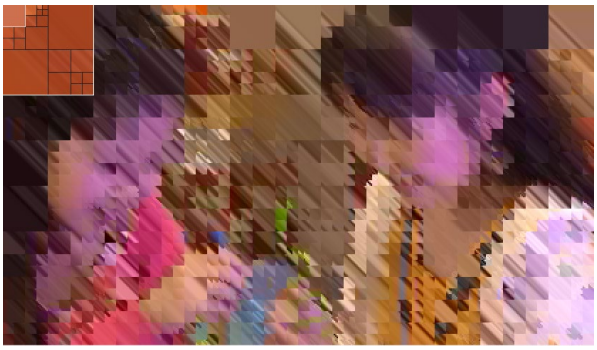

图 3 全部使用角度模式 18 进行预测

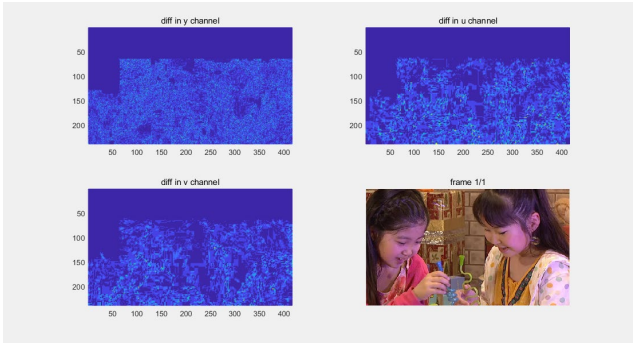

图 6 从第 9 个 LCU 引入失真

2.2 失真的传递

依据 HEVC 标准，对于任意帧内块的预测都需要依赖于其周边相邻块的重建像素（如果存在的话）。

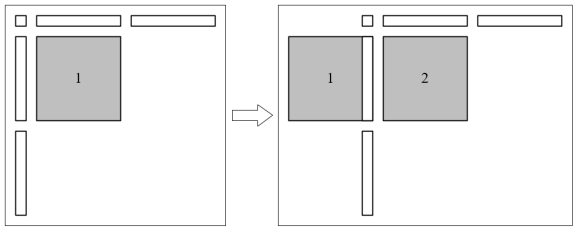

图 4 预测的示意图

如上图所示，对于像素块 1 和像素块 2 的预测（标记为灰色），都需要依赖于其左上，上方，右上，左方和左下的重建像素（标记为白色）。而显然地，像素块 2 所需要的左方参考像素恰恰就是像素块 1 经过预测、求差、变换、量化、反量化、反变换和求和所得到的重建像素。因此，如果由于预测方式的变化在块 1 中引入了失真，那么该失真很可能会被传播给块 2，并继续传播给后续的图像块。

下图分别展示了从第 1 个，第 9 个和第 17 个 LCU 引入失真后的传递情况。

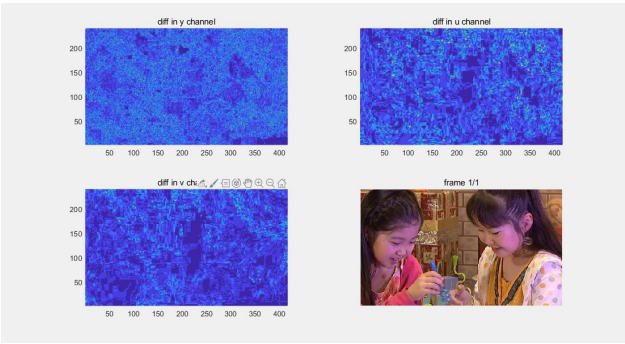

图 5 从第 1 个 LCU 引入失真

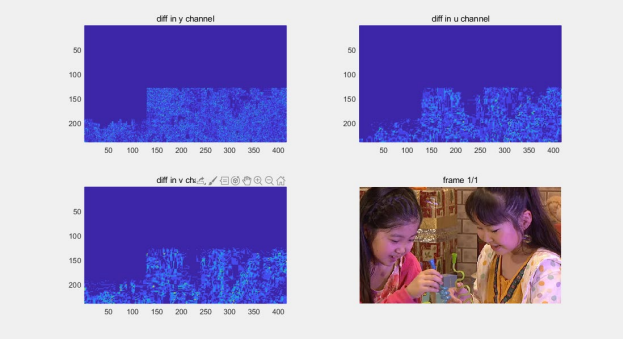

图 7 从第 17 个 LCU 引入失真

图中的深蓝色表示没有差异，浅蓝色表示有一定的差异，其他颜色表示差异较大。

2.3 失真的控制

尽管为了引入失真，该算法改变了某个块所选出的最佳方向，但是其余块的最佳方向仍然是依照编码器正常的代价函数来进行选择的。因此，后续的块仍然能够保证一定的率失真水平。

以下基于 x265，对该算法进行了测试。

调用 x265 时所采用的参数如下：

```
--input          $FILE_SEQ.yuv
--fps            $DATA_FPS
--input-res      ${SIZE_FRA_X}x${SIZE_FRA_Y}
--input-depth    $DATA_PXL_WD
--frames         $NUMB_FRA
--keyint         $SIZE_GOP
--qp             $DATA_QP
--ipratio        1
--pbratio        1
--output         ${PREF_DMP}x265.h265
--output-depth   $DATA_PXL_WD
```

```
--recon          ${PREF_DMP}x265.yuv
--recon-depth    $DATA_PXL_WD
--psnr
--tune           psnr
--log-level      full
--no-progress
--frame-threads
--no-wpp
--no-pmode
--no-pme
```

其中所包含变量的意义如下：

表 1 所包含变量的意义

变量	含义	附注
FILE_SEQ	序列的位置	
DATA_FPS	序列的帧率	
SIZE_FRA_X	序列的宽度	
SIZE_FRA_Y	序列的高度	
DATA_PXL_WD	序列的比特深度	
NUMB_FRA	编码的帧数	
SIZE_GOP	编码的 GOP	为了简化比较，此处仅测试了帧内编码，因此，该值设为 1
DATA_QP	编码的量化参数	为了得到 B-D 码率的结果，此处遍历了量化参数 22，27，32，37
PREF_DMP	导出的位置	

经测试，B-D 码率的结果如下：

表 2 原算法的 B-D 码率情况

序列	Y	U	V	平均
BasketballPass	0.9%	0.4%	0.9%	0.8%
BQSquare	0.8%	0.5%	0.1%	0.6%
BlowingBubbles	0.4%	0.0%	0.0%	0.3%
RaceHorses	0.4%	0.3%	0.2%	0.4%
BasketballDrill	1.3%	1.0%	1.1%	1.2%
BQMall	1.0%	0.7%	0.7%	0.9%
PartyScene	0.6%	0.3%	0.3%	0.5%
RaceHorsesC	0.6%	0.3%	0.4%	0.5%
FourPeople	0.8%	0.3%	0.5%	0.7%
Johnny	1.2%	0.7%	0.7%	1.0%
KristenAndSara	1.0%	0.8%	1.0%	1.0%
Kimono	0.4%	0.2%	0.2%	0.3%
ParkScene	0.5%	0.2%	0.1%	0.4%
Cactus	0.8%	0.3%	0.4%	0.6%
BasketballDrive	1.1%	0.8%	0.7%	1.0%
BQTerrace	0.8%	0.5%	0.5%	0.7%
Traffic	0.7%	0.5%	0.3%	0.6%
PeopleOnStreet	0.8%	0.6%	0.3%	0.7%

换言之，上述水印平均引入的 B-D 码率增量仅为 0.7%。

3. 原始算法

该论文的具体算法如下：

选择某个特定块，

（1） 如果编码器所选择的最佳模式为 0，则将其修改为 1；

（2） 如果编码器所选择的最佳模式为 1，则将其修改为 0；

（3） 如果编码器所选择的最佳模式为 17 或 34，则将其修改为 16 或 33；

（4） 如果编码器所选择的最佳模式为其余模式，则将其修改为原模式加 1。

对应的伪代码如下：

伪代码 1. 原算法

输入：原始模式

输出：最终模式

```
1.  if (idxBlk == IDX_BLK_TARGET) {
2.      if (mode == 0)
3.          mode = 1;
4.      else if (mode == 1)
5.          mode = 0;
6.      else if (mode == 17 || mode == 34)
7.          --mode;
8.      else
9.          ++mode;
10. }
```

其中，mode 代表编码器所选定的模式，idxBlk 代表当前块的索引，IDX_BLK_TARGET 代表目标块的索引。

显然，该算法的主要优点，或者说创新点，在于对编码器特性的高效利用。具体而言，该算法，

（1）一方面利用了编码器对于数据的强依赖特性，将局部的扰动扩散成了能够被识别的水印信息；

（2）另一方面又利用了编码器本身的代价函数，将水印信息对于视频率失真水平的影响限制在了非常小的范围内。

然而，该算法也并非尽善尽美，至少存在以下缺点：

（1）该算法并未提供选择特定块的有效方式，如果选择不当，可能无法造成失真，或者失真无法被扩散成能够被识别的水印信息。

（2）该算法选择替换模式的依据并不充分，尤其是对 Planar 模式和 DC 模式进行的对调。

（3）该算法的输入参数只有位置，难以满足为多个用户加入多个水印的需求。

（4）该算法并未提供调节水印“强度”的有效方式，基于该算法，似乎只能通过选择位置改变水印的覆盖范围，但并不能改变水印的具体“强度”。

4. 算法改进

4.1 改进一

针对第一个缺点：“该算法并未提供选择特定块的有效方式，如果选择不当，可能无法造成失真，或者失真无法被扩散成能够被识别的水印信息”，容易想到的改进是：将原算法从“选择某个特定块”修改为“对于所有的块”。对应的伪代码如下：

伪代码 2. 改进一

输入：原始模式

输出：最终模式

```
1.  if (mode == 0)
2.      mode = 1;
3.  else if (mode == 1)
4.      mode = 0;
5.  else if (mode == 17 || mode == 34)
6.      --mode;
7.  else
8.      ++mode;
```

显然，修改后的算法不仅可以确保造成失真，而且能够确保失真分布在整个画幅内。然而，这也意味着水印很可能会急剧地影响编码的质量。因此，以下对修改后算法的 B-D 码率进行了测试，具体的结果如下：

表 3 改进一的 B-D 码率情况

序列	Y	U	V	平均
BasketballPass	25.1%	15.8%	16.2%	22.1%
BQSquare	18.1%	10.8%	10.7%	15.7%
BlowingBubbles	17.3%	8.3%	8.2%	14.3%
RaceHorses	17.7%	11.0%	11.4%	15.6%
BasketballDrill	31.4%	24.3%	25.5%	29.2%
BQMall	24.9%	15.2%	14.9%	21.7%
PartyScene	16.8%	9.6%	9.3%	14.4%
RaceHorsesC	16.1%	8.3%	8.6%	13.5%
FourPeople	27.7%	14.5%	13.2%	23.1%
Johnny	32.0%	22.7%	21.9%	28.8%
KristenAndSara	29.3%	19.6%	17.7%	25.8%
Kimono	11.2%	5.9%	5.9%	9.4%
ParkScene	13.2%	4.0%	3.3%	10.0%
Cactus	19.9%	11.0%	11.7%	17.0%
BasketballDrive	28.8%	15.4%	16.7%	24.5%
BQTerrace	20.2%	11.0%	10.3%	17.0%
Traffic	19.0%	11.4%	10.2%	16.3%
PeopleOnStreet	22.0%	13.6%	11.7%	18.9%

根据上述结果可知，修改后的 B-D 码率从原先的 0.7%恶化到了 18.7%。这一恶化在实际的工程应用中是难以接受的，因此，我们需要进一步的改进。

4.2 改进二

值得注意的是，上述结果也从另一个侧面反映了该算法的第二个缺点：“该算法选择替换模式的依据并不充分，尤其是对 Planar 模式和 DC 模式进行的对调。”因此，改进二尝试了从替换模式的角度去尝试优化 B-D 码率。

事实上，编码器选择模式的过程就是按照代价函数对各个模式进行排序的过程。因此，与其选择交换 DC 模式和 Planar 模式，对当前角度模式自增一或者对当前模式自减一，不如选择排序后的次优模式作为替换。对应的伪代码如下：

伪代码 3. 改进二

输入：原始模式

输出：最终模式

```
1.  mode = modeList[1];
```

其中，modeList 为排序后的模式。

各个序列的具体表现如下所示：

表 4 改进二的 B-D 码率情况

序列	Y	U	V	平均
BasketballPass	15.9%	11.2%	11.2%	14.3%
BQSquare	13.5%	8.9%	8.5%	11.9%
BlowingBubbles	11.6%	6.3%	6.0%	9.8%
RaceHorses	13.3%	8.9%	8.9%	11.8%
BasketballDrill	21.0%	16.3%	16.9%	19.5%
BQMall	17.9%	11.5%	11.5%	15.8%
PartyScene	11.3%	6.9%	6.9%	9.9%
RaceHorsesC	11.5%	6.3%	6.5%	9.8%
FourPeople	19.9%	10.9%	10.4%	16.9%
Johnny	25.0%	17.9%	17.0%	22.5%
KristenAndSara	22.6%	15.4%	14.1%	20.0%
Kimono	9.5%	5.0%	5.0%	8.0%
ParkScene	9.2%	2.5%	2.1%	6.9%
Cactus	14.3%	8.0%	8.5%	12.3%
BasketballDrive	17.2%	8.1%	9.2%	14.3%
BQTerrace	13.9%	8.0%	7.6%	11.9%
Traffic	13.7%	8.3%	7.2%	11.7%
PeopleOnStreet	16.9%	10.4%	8.8%	14.4%

根据上述结果，尽管修改替换方式后的平均 B-D 码率的确优于原替换方式的 18.7%，但仍然高达 13.4%，无法为实际工程应用所接受，需要进一步改进。

4.3 改进三

为了进一步降低 B-D 码率，我引入了一个异或链（此处直接使用伪随机数发生器实现），当该异或链的输出（单比特）为 1 时，引入失真，否则保持原始值。事实上，通过设置该异或链的初始值（此处直接使用 srand 实现），我们还可以解决第三个缺点，即：“该算法的输入参数只有位置，难以满足为多个用户加入多个水印的需求。”对应的伪代码如下：

伪代码 4. 改进三

输入：原始模式

输出：最终模式

```
1.  static bool flgIni = 1;
2.  if (flgIni) {
3.      flgIni = 0;
4.      srand(DATA_SEED);
5.  }
6.  if (rand() % 2) {
7.      mode = modeList[1];
8.  }
```

以下仍然对 B-D 码率结果进行测试，具体的结果如下：

表 5 改进三的 B-D 码率情况

序列	Y	U	V	平均
BasketballPass	6.9%	4.6%	4.6%	6.1%
BQSquare	6.4%	4.5%	4.0%	5.7%
BlowingBubbles	5.4%	3.1%	3.0%	4.6%
RaceHorses	5.9%	4.0%	4.0%	5.2%
BasketballDrill	8.8%	6.6%	7.1%	8.1%
BQMall	8.0%	5.1%	5.1%	7.0%
PartyScene	5.3%	3.3%	3.3%	4.6%
RaceHorsesC	5.3%	2.7%	2.7%	4.4%
FourPeople	8.6%	4.6%	4.2%	7.2%
Johnny	10.1%	6.7%	6.5%	8.9%
KristenAndSara	9.6%	6.4%	5.9%	8.5%
Kimono	4.4%	2.4%	2.3%	3.7%
ParkScene	4.7%	1.3%	0.9%	3.5%
Cactus	6.5%	3.5%	3.7%	5.5%
BasketballDrive	7.4%	3.1%	3.2%	5.9%
BQTerrace	5.7%	3.1%	2.8%	4.8%
Traffic	6.2%	3.6%	3.0%	5.2%
PeopleOnStreet	7.5%	4.6%	3.9%	6.4%

根据上述结果，引入异或链后的 B-D 码率已经低至 5.9%，勉强达到可以使用的程度。除此之外，由于异或链的引入，发行者可以针对不同的用户选用不同的初始值，从而达到甄别泄露源的目的。

4.4 改进四

为了解决第四个问题：“该算法并未提供调节水印“强度”的有效方式，基于该算法，似乎只能通过选择位置改变水印的覆盖范围，但并不能改变

水印的具体“强度”。”，我不再使用异或链的输出作为替换模式的使能，而是根据内部状态（，如，是否小于某个特定值，）来决定是否替换模式。一种可行的伪代码如下：

伪代码 5. 改进四

```
1.  static bool flgIni = 1;
2.  if (flgIni) {
3.      flgIni = 0;
4.      srand(DATA_SEED);
5.  }
6.  if (rand() % 100 < DATA_RATE) {
7.      mode = modeList[1];
8.  }
```

上述代码约束了大约 DATA_RATE%的块会被替换最佳模式。换言之，调节 DATA_RATE，就可以对水印“强度”和编码效果进行权衡。经测试，当 DATA_RATE 等于 6，即 6%时，能够取得略优于原算法持平的 B-D 码率表现，具体结果如下所示：

表 6　改进四的 B-D 码率情况

序列	Y	U	V	平均
BasketballPass	0.7%	0.5%	1.0%	0.7%
BQSquare	0.7%	0.7%	0.2%	0.6%
BlowingBubbles	0.5%	0.4%	0.3%	0.5%
RaceHorses	0.7%	0.4%	0.3%	0.6%
BasketballDrill	0.9%	0.7%	0.9%	0.9%
BQMall	0.9%	0.7%	0.8%	0.8%
PartyScene	0.6%	0.4%	0.3%	0.5%
RaceHorsesC	0.6%	0.3%	0.3%	0.5%
FourPeople	0.9%	0.5%	0.6%	0.8%
Johnny	1.1%	0.3%	0.9%	0.9%
KristenAndSara	1.0%	0.9%	0.6%	0.9%
Kimono	0.5%	0.2%	0.2%	0.4%
ParkScene	0.5%	0.2%	0.2%	0.4%
Cactus	0.7%	0.3%	0.6%	0.6%
BasketballDrive	0.8%	0.5%	0.5%	0.7%
BQTerrace	0.6%	0.4%	0.2%	0.5%
Traffic	0.7%	0.4%	0.3%	0.6%
PeopleOnStreet	0.8%	0.5%	0.4%	0.7%

4.5 测试与总结

原算法的作者将某个块的模式变化在后续块所引入（隐式的）失真视作为水印信息；而本文则是将对于各个块的扰动情况（即扰动和不扰动）视作为水印信息。具体而言，相比于原算法，改进后的算法选择了一种更为合理的（，或者说，失真更小的）替换方式，并根据异或链的状态有约束地施加到视频的整个画幅上，因此，直接带来的好处有：

（1） 水印的分布更平均，不用担心由于对替换块的选择不当，而无法造成失真或者失真无法被扩散成能够得到识别水印信息。

（2） 替换模式的依据更合理，因此水印的强度更平均。

（3） 可以通过指定异或链的初始状态为多个用户加入多个不同的水印。

（4） 可以根据需求调节被替换块的比率，从而调节水印的“强度”。

以下测试了设置不同初始状态后，与原图的差异情况：

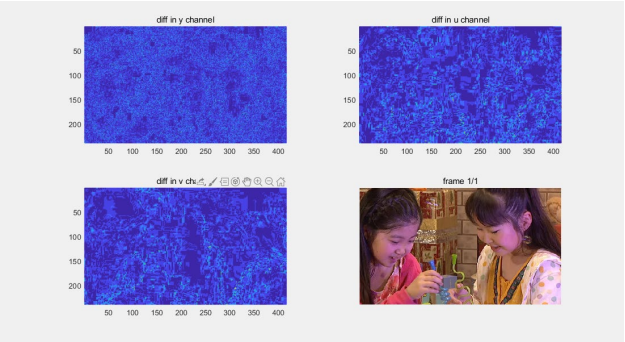

图 6　初始状态为 1

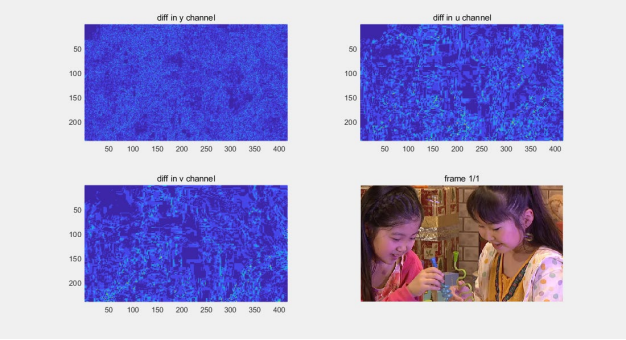

图 6　初始状态为 2

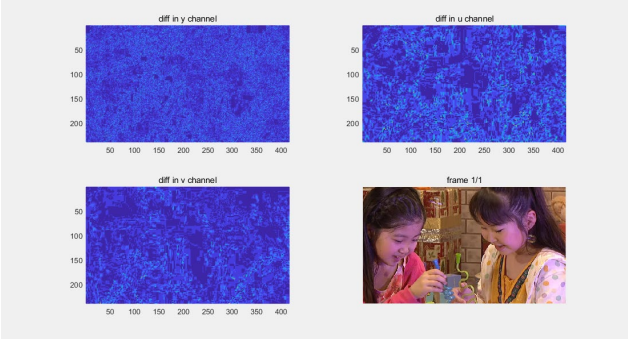

图 7　初始状态为 3

图中的深蓝色表示没有差异，浅蓝色表示有一定的差异，其他颜色表示差异较大。

显然，改进后算法的差异更平均，而原算法的差异直接与替换块的位置紧耦合，如图 5-7 所示：

以下测试了经过重编码攻击后的提取情况：

重编码的方法是：

（1）对所有加入水印后的视频进行解码得到 YUV 数据

（2）对上述所有的 YUV 数据（以同样的 QP）进行编码

提取的方法是：

（1）对所有重编码的视频进行解码得到 YUV 数据

（2）对所有加入水印后的视频进行解码得到 YUV 数据

（3）计算两两之间的 PSNR

以下分别采用原算法和改进后算法为同一序列添加了两组不同的水印（每组 10 个），并按照上述重编码和提取的方法进行处理，并得到下图：

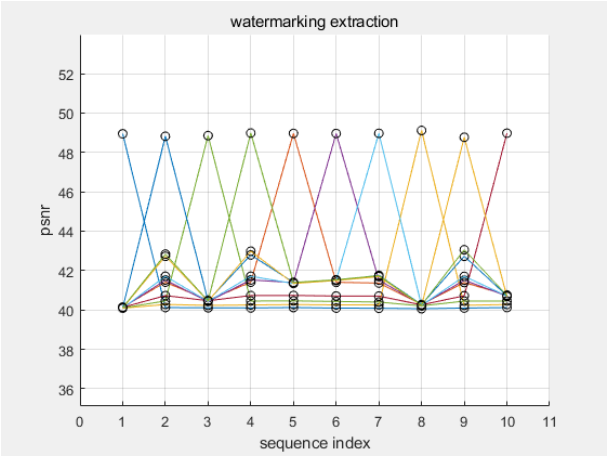

图 8　原始算法的提取情况

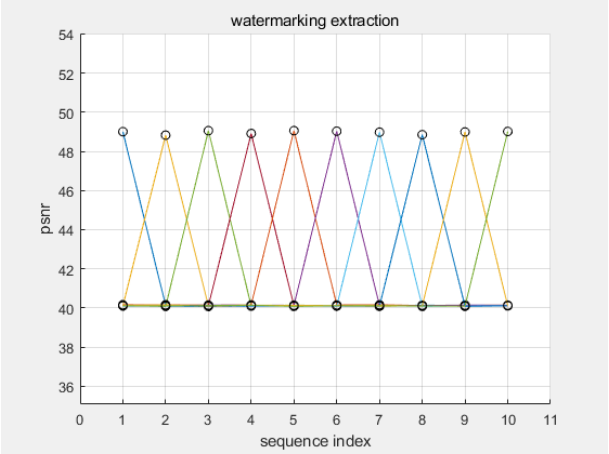

图 9　改进算法的提取情况

显然，改进后算法的稳定度更高，而原算法则不太稳定。

4.6 后续工作

由于时间原因，本次调研的测试较为简单：

（1）只测试了帧内帧的情况

（2）只测试了以相同 QP 重编码的攻击

应在后续的工作中补全更多的测试和分析。

另外，基于本文的这一想法：“对于各个块的扰动情况（即扰动和不扰动）视作为水印信息。”，改进后算法可以被进一步地扩展，例如：

（1）除了扰动帧内预测模式之外，也可以扰动帧间预测矢量，或者编码 QP，这同样能够起到添加水印的目的。

（2）除了调节替换块的比例之外，也可以调节扰动的强度，这同样能够起到改变水印强度的目的。

参 考 文 献

[1] H. Mareen, J. De Praeter, G. Van Wallendael and P. Lambert, "A Novel Video Watermarking Approach Based on Implicit Distortions," in IEEE Transactions on Consumer Electronics, vol. 64, no. 3, pp. 250-258, Aug. 2018, doi: 10.1109/TCE.2018.2852258.

[2] S. Gaj, A. Sur, and P. K. Bora, “A robust watermarking scheme against re-compression attack for H.265/HEVC,” presented at the 5th Nat. Conf. Comput. Vis. Pattern Recognit. Image Process. Graph., Patna, India,2015, pp. 1–4

[3] T. Dutta and H. P. Gupta, “A robust watermarking framework for high efficiency video coding (HEVC)-Encoded video with blind extraction

process,” J. Vis. Commun. Image Represent., vol. 38, pp. 29–44,Jul. 2016

[4] T. Kalker, G. Depovere, J. Haitsma, and M. J. Maes, “Video watermarking system for broadcast monitoring,” inProc. SPIESecurity Watermarking Multimedia Contents, vol. 3657. San Jose, CA, USA, 1999, pp. 103–112

[5] F. Hartung and B. Girod, “Watermarking of uncompressed and compressed video,” Signal Process., vol. 66, no. 3, pp. 283–301, 19

[6] T. Yamada, M. Maeta, and F. Mizushima, “Video watermark application for embedding recipient ID in real-time-encoding VoD server,” J. Real Time Image Process., vol. 11, no. 1, pp. 211–222, 20