基于压缩感知的视频双水印算法研究

周 燕[1] 曾凡智[1] 赵慧民[2]
(佛山科学技术学院计算机系 佛山 528000)[1] (广东技术师范学院电子与信息学院 广州 510665)[2]

摘 要 针对数字视频的内容保护与帧内、帧间篡改检测的难题,采用压缩感知理论提取视频的内容特征作为水印,提出一种双水印的视频保护和篡改检测算法。首先,利用压缩感知过程提取I帧宏块的内容特征,生成半脆弱的内容认证水印;然后,对帧序号进行二值运算,生成完整性水印;最后,利用压缩感知信号重构OMP(Orthogonal Matching Pursuit)算法把生成的双水印嵌入到I帧和P帧相应宏块的DCT高频系数的压缩测量值中,以此提高视频水印的抗攻击能力,并实现对视频篡改的检测。仿真实验表明,所提算法对视频帧内篡改具有精确定位到子块的检测能力;同时对帧插入、帧删除、帧交换等类型的视频帧间篡改具有很强的检测能力。

关键词 压缩感知,内容认证水印,完整性水印,帧内篡改检测,帧间篡改检测
中图法分类号 TP391 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.5.025

Double Video Watermarking Algorithm Based on Compressive Sensing

ZHOU Yan[1] ZENG Fan-zhi[1] ZHAO Hui-min[2]
(Department of Computer Science, Foshan University, Foshan 528000, China)[1]
(School of Electronic and Information, Guangdong Polytechnic Normal University, Guangzhou 510665, China)[2]

Abstract For the problem of video content protection and tamper detection in frame or between frames, by extracting the video content features as watermarking based on compressive sensing(CS) theory, we proposed a video protection and tamper detection algorithm based on double watermarking. Firstly, the content features of the macro blocks of I frame are extracted by CS procedure, and the semi-fragile authentic watermark is generated. Then, by performing binary arithmetic to the frame sequences, the integrity watermark is generated. Finally, the double watermarking is embedded into the CS measurements of high-frequency DCT coefficients in macro blocks of I frame and P frame by the OMP procedure of CS reconstruction. This algorithm can improve the anti-attack capability of video watermarking, and detect the tamper to video. Simulation results show that the proposed algorithm has accurate detecting capability for tamper in frame, which can detect the sub-blocks of video frame. Also, it has precise detecting capability for tamper between frames such as frame insertion, frame deletion and frame switch, which can detect any tampered frame.

Keywords Compressive sensing, Authentic watermarking, Integrity watermarking, Tamper detect in frame, Tamper detect between frames

1 引言

数字视频技术在平安城市、智能交通、新闻媒体等领域获得了广泛应用。但大量编辑工具的出现,也导致视频篡改、视频伪造引发的恶性事件频发。视频数据的复杂性和格式的多样性,使得视频攻击形式多样,出现了帧修改、帧编辑等针对帧内容的帧内篡改,以及帧插入、帧删除、帧交换等针对帧连续性的帧间篡改等攻击手段。如何保证数字视频安全可信,是一项亟待解决的挑战性问题[1]。针对图像的篡改检测研究已相对成熟,而针对视频的篡改检测方法相对较少。

近年来,数字水印技术成为视频保护的重要手段之一,是视频防伪与安全领域的一个研究热点[2]。文献[3]针对数字视频的剪辑篡改提出一种对拼接的盲检测方法,该方法对变化比较平缓的视频序列的检测效果较好。文献[4]和文献[5]提出了一种视频Hash水印及篡改检测方法,其能较好地检测出视频帧间篡改,但数字签名会增加视频的传输数据量。视频水印的嵌入方式可分为在原始视频中嵌入、在编码压缩域中嵌入、在已压缩视频中嵌入3种主要模式。随着各类视频压缩格式的推广应用,压缩域中的视频水印技术成为研究重点,大量研究选择在视频DCT系数中嵌入数字水印[6-8]。文献[6]采用运动矢量与DCT相结合的水印算法实现帧内篡改检测,但对帧间篡改的检测能力较弱。文献[7]针对视频帧内篡改的检测,提出了一种针对MPEG-2恶意篡改的可逆半脆弱水印算法,该算法将水印嵌入视频帧的分块DCT系数

到稿日期:2015-10-26 返修日期:2015-12-03 本文受国家自然科学基金(61272381),广东省自然科学基金项目(2015A030313635,2015A030313672),广东省科技计划项目(2014A010103037),佛山市科技创新专项资金项目(2015AG10008,2014AG10001),广东省教育厅特色创新类项目(2015KTSCX153),佛山科学技术学院优秀青年教师培养计划项目(fsyq201411),佛山科学技术学院优秀青年人才培育项目资助。
周 燕(1979-),女,硕士,副教授,主要研究方向为图像处理、信息安全,E-mail:zhouyan791266@163.com;曾凡智(1965-),男,博士,教授,主要研究方向为图像处理、数据挖掘;赵慧民(1966-),男,博士,教授,主要研究方向为图像处理、信息安全。

中，实现帧内篡改定位，但定位精度有待改善，并且不适用于帧间篡改检测。文献[8]提出了一种适用于 MPEG-4 内容认证的半脆弱可逆视频水印算法，该算法在 I 帧中嵌入双水印，实现对视频内容的认证和篡改定位，但对于帧插入和帧删除的帧间篡改定位精度有待提高。

最近，有研究者提出在视频压缩域中同时嵌入帧内容认证水印与帧间完整性水印的双水印技术[9-11]。文献[9]提出了一种结合 I 帧分区特征提取和 ECC 编码的认证水印与完整水印算法，该算法对帧插入和帧删除等帧间篡改的定位比较准确，但同时增加了水印长度。文献[10]提出了一种基于 DCT 系数分组统计特征的认证水印和基于帧序列的完整水印算法，该算法对帧内篡改的检测效果较好，而对帧间篡改的检测过于依赖 DCT 对角系数。文献[11]提出了一种提取运动对象的双水印算法，该算法对运动对象明显的视频的篡改检测效果较好，而对于运动对象较少的视频，其水印嵌入强度受到一定程度的影响。把 CS 理论应用到数字水印方面，成为一种新的研究手段[12,13]。文献[12]中较早地提出了利用 CS 抽取图像的内容特征，从而生成内容认证水印的算法。文献[13]论证了 CS 测量值的安全性，并证明了测量值具有计算保密性，为在测量值上嵌入数字水印提供了理论基础。有研究者把 CS 引入到视频水印方面，并且取得了初步进展[14,15]。文献[14]中提出利用 CS 理论分析帧内容特征，以生成认证水印，采用 HASH 运算生成完整性水印，实现了帧内及帧间篡改检测，但尚未把 CS 理论应用到完整性水印方面。文献[15]采用 CS 测量过程生成内容认证水印并实现了视频的帧内篡改检测的方法，但未涉及帧间篡改检测。

针对目前的视频水印算法在篡改检测方面存在的不足，需要利用 CS 理论中关于压缩测量值的保密性，在认证水印与完整性水印的双水印算法方面做进一步的研究。本文以视频结构及 I 帧编码方式为基础，首先通过 DCT 变换获取宏块中各子块的频域系数，并结合压缩感知理论，抽取宏块图像的内容特征，生成 I 帧的内容认证水印；接着对 I 帧和 P 帧序列进行二值编码，得到视频完整性水印；最后，采用 OMP 信号重构算法，把认证水印嵌入到亮度子块的 DCT 高频系数的压缩测量值中，把完整性水印嵌入到色度子块与 P 帧亮度子块的压缩测量值中。通过提取认证水印和完整性水印并进行对比，实现对 I 帧的帧内篡改检测和帧间篡改检测。

2 压缩感知与测量矩阵

压缩感知理论指出：由少量压缩测量值，能够精确地恢复出可稀疏的原始信号。利用这一特性可以把 CS 应用于数据降维与特征提取等方面。设 $X\in R^N$ 是长度为 N 的一维信号，若 $SUP(X)=\{i|X_i\neq 0\}$ 且满足 $\#(SUP(X))\leqslant K$ 时，称 X 为 K-稀疏信号。选取大小为 $M\times N(M\ll N)$ 的压缩测量矩阵 Φ，采用如下测量过程可以得到长度为 M 的测量向量 $Y\in R^M$：

$$Y=\Phi X \tag{1}$$

对于 K-稀疏信号，当测量矩阵 Φ 满足 RIP(Restricted Isometry Property)条件，并且 RIP 系数 δ_k 满足 $\delta_{2k}\leqslant\sqrt{2}-1$，测量数目 M 满足 $M\geqslant c(K\log N)$ 条件时，少量测量值 Y 能够精确重构原始信号 X。X 的求解与优化问题(2)等价[16]：

$$\min_X \| X \|_1 \quad \text{s.t. } Y=\Phi X \tag{2}$$

对凸优化问题(2)的求解已有许多经典算法，包括 MP、OMP、SAMP 等[17-19]。正交匹配追踪 OMP 算法在 MP 算法的基础上采用正交化技术，以较少的迭代次数能够快速重构稀疏的原始信号。本文把 CS 应用于视频 I 帧的内容认证水印生成和嵌入过程中，重点考虑视频 I 帧的宏块中 8×8 子块的 DCT 系数高频部分，其高频系数具有很好的稀疏性，通过信号重构完成水印的嵌入过程。所采用的 OMP 算法框架如下。

算法 1 OMP 算法

输入：测量矩阵 Φ，向量 Y，最大稀疏度 s

输出：恢复的信号 X^*

初始值：$R_0=Y, X_0=0, \Lambda_0=\varnothing, k=0$

过程：

While $k\leqslant s$

残差投影：$H_k=\Phi^T\cdot R_k$

追踪最大分量及更新支撑集：

$\Lambda_{k+1}=\Lambda_k\cup\{\arg\max_j|H_k(j)|\}$

最小二乘投影：

$X_{k+1}=\arg\min_{Z:SUP(Z)\subseteq\Lambda_{k+1}}\| Y-\Phi Z\|_2$

计算新残差：$R_{k+1}=Y-\Phi\cdot X_{k+1}$

$k=k+1$

End while

$X^*=X_{s+1}$

原始信号的重构效率依赖于压缩测量矩阵 Φ，把重构过程记为 $X^*=OMP(Y,\Phi)$。在 CS 中通常选取 Gauss 随机矩阵作为压缩感知的测量矩阵，它能以高概率满足 RIP 条件，对于长度为 N、稀疏度为 K 的信号，$M\geqslant cK\log(N)$ 个测量值就可以高概率地重构出原始信号[16]。本文选取测量矩阵 $\Phi=(\Phi_{i,j})_{M\times N}$，其矩阵元素 $\Phi_{i,j}$ 服从均值为零、方差为 $1/\mathrm{sqrt}(M)$ 的正态分布，并且元素间相互独立，即：

$$\Phi_{i,j}\sim N(0,1/\sqrt{M})$$

最后将所产生的高斯随机矩阵的每一列进行归一化处理后的结果作为本文的测量矩阵。

3 基于双水印的视频篡改检测框架

3.1 视频结构分析

标准视频是以包含 I 帧和 P 帧的图像组为基础，以帧内宏块与子块为层次，形成结构化和层次化的视频压缩结构。为了实现视频帧内认证水印与帧间完整性水印的生成与嵌入，本文以 MPEG-2、MPEG-4 等为研究对象。设视频帧序列如下：

$$I_1P_{11}P_{12}\cdots P_{1m_1}\cdots I_iP_{i1}P_{i2}\cdots P_{im_i}\cdots I_nP_{n1}P_{n2}\cdots P_{nm_n}$$

$I_iP_{i1}P_{i2}\cdots P_{im_i}$ 表示第 i 个图像组(GOP)中的 I 帧以及 m_i 个 P 帧图像，n 表示视频的 GOP 数量，视频 I 帧与 P 帧的总数 N_F 满足如下关系：

$$N_F=\sum_{i=1}^{n}(m_i+1) \tag{3}$$

视频 I 帧包含视频的关键信息，是运动 P 帧的参考基准，内容认证水印主要针对 I 帧的内容进行保护。视频 I 帧可表示为：

$$I=\{I_1,I_2,\cdots,I_n\}$$

设每幅 I 帧图像的大小为 $N_1\times N_2$，宏块大小为 $B\times B$，其中 B 能整除 N_1、N_2。视频中第 i 个 I 帧的宏块集如下(按行优先次序)：

$$MB_i=\{MB_{i1},MB_{i2},\cdots,MB_{ib}\}$$

其中 $b=N_1\times N_2/B^2$。

假设视频采样比例为 4:2:0，选取 $B=16$，则每个宏块 MB_{ij} 包含 4 个亮度子块和 2 个色度子块，每个子块的大小为 8×8。不失一般性，设 MB_{ij} 的组成如下：

$$MB_{ij}=\{B_1,B_2,B_3,B_4,C_1,C_2\}$$

其中，前 4 个代表亮度块，后 2 个代表色度块。

3.2 基于双水印的视频篡改检测框架

本文提出的基于双水印技术的视频篡改检测方法是在编码 DCT 系数的压缩测量中嵌入水印，其整体框架如图 1 所示，主要分为视频保护和篡改检测两部分。对视频的保护主要基于压缩感知的视频内容认证水印和完整性水印，对视频篡改的检测包括帧间篡改检测和帧内篡改检测，最后对视频篡改进行综合分析并输出检测结果。

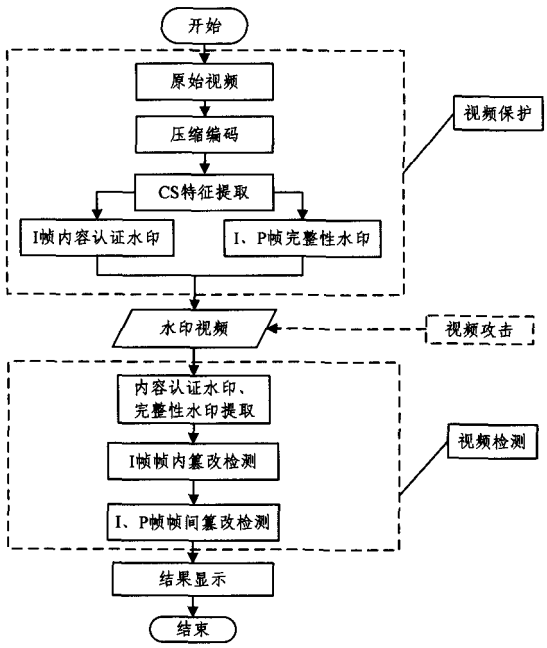


图 1 基于 CS 的视频双水印嵌入与篡改检测框架

4 基于 CS 的视频内容认证水印

4.1 视频内容认证水印的生成与嵌入

内容认证水印是根据视频 I 帧的内容特征生成水印，目标是对 I 帧图像常见的图像更替、擦除等篡改方式进行检测。视频 I 帧的宏块集中包含 4 个亮度子块和 2 个色度子块，亮度子块反映宏块内的主要内容特征，对视频帧内的内容篡改将会改变亮度子块的 DCT 中、低频系数。因此，将亮度子块的 DCT 中、低频系数作为内容特征，同时因系数在压缩过程中相对稳定，由此生成的认证水印不仅可以保证其鲁棒性，还可以利用其检测帧内攻击和篡改。利用文献[13]中指出的在 CS 压缩测量值中嵌入水印具有计算保密性(Computational Secrecy)特征，采用本文第 2 节描述的 OMP 算法，把认证水印嵌入到亮度子块高频系数的压缩测量值中。内容认证水印的生成与嵌入过程如图 2 所示。

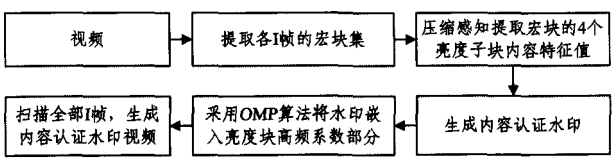


图 2 内容认证水印的生成与嵌入过程

视频内容认证水印的生成与嵌入算法过程如下。

Step1 按照第 2 节描述的方式，选取 Gauss 随机测量矩阵 $\Phi\in R^{M\times 32}$，矩阵 Φ 作为密钥参与水印提取过程，一般选取 $M\geqslant 6$。

Step2 提取视频 I 帧的宏块 MB_{ij}，用 B_1,B_2,B_3,B_4 表示一个宏块内的 4 个亮度子块(按从左到右、从上到下进行排序)，每个子块的大小为 8×8，通过 DCT 变换得到子块的 DCT 系数。

Step3 对 B_1 中 8×8 的 DCT 系数，按 Zig-Zag 扫描的顺序形成 64 维的系数向量 $Z^{B_1}\in R^{64}$。在此基础上抽取向量 $X^{B_1},\overline{X}^{B_1}\in R^{32}$，它们分别代表亮度子块的中低频、高频系数部分。其中：

$$X^{B_1}(i)=Z^{B_1}(i),\overline{X}^{B_1}(i)=Z^{B_1}(i+32)$$

其中，$i=1,2,\cdots,32$。

Step4 采用压缩测量式(1)对向量 X^{B_1}、$\overline{X}^{B_1}$ 分别进行压缩测量：

$$Y^{B_1}=\Phi\cdot X^{B_1},\overline{Y}^{B_1}=\Phi\cdot\overline{X}^{B_1} \tag{4}$$

对 B_2、B_3、B_4 进行同样的处理，得到的压缩测量向量分别记为：

$$Y^{B_2},\overline{Y}^{B_2},Y^{B_3},\overline{Y}^{B_3},Y^{B_4},\overline{Y}^{B_4}$$

Step5 生成基于内容特征的水印。4 个亮度子块 B_1、B_2、B_3、B_4 对应的内容认证水印记为：$w_i\in\{0,1\}$，$i=1,2,3,4$。生成公式如下：

$$w_i=\begin{cases}1, & \text{if}\ \|Y^{B_i}-\tilde{Y}\|_2\geqslant T\cdot D\\ 0, & \text{else}\end{cases} \tag{5}$$

其中，$\tilde{Y}=\frac{1}{4}\sum_{k=1}^{4}(Y^{B_k})$ 代表特征向量的中值，$D=\frac{1}{4}\sum_{k=1}^{4}\|Y^{B_k}\|_2$ 代表一个宏块内 4 个亮度子块的平均强度，T 代表水印生成阈值，本文取 $T=0.15$。

Step6 嵌入内容认证水印。按式(6)、式(7)在亮度子块 B_i 的高频系数压缩测量值 $\overline{Y}^{B_i}$ 上嵌入水印，每个水印被重复嵌入 3 次，以保证水印的鲁棒性。

$$\text{if}((w_i==1)\ \&\&(\overline{Y}^{B_i}(2k)\leqslant\overline{Y}^{B_i}(2k-1)))$$

$$\begin{cases}\overline{Y}^{B_i}(2k)=\overline{Y}^{B_i}(2k-1)+\alpha\\ \overline{Y}^{B_i}(2k-1)=\overline{Y}^{B_i}(2k)-\alpha\end{cases} \tag{6}$$

$$\text{if}((w_i==0)\ \&\&(\overline{Y}^{B_i}(2k)\geqslant\overline{Y}^{B_i}(2k-1)))$$

$$\begin{cases}\overline{Y}^{B_i}(2k)=\overline{Y}^{B_i}(2k-1)-\alpha\\ \overline{Y}^{B_i}(2k-1)=\overline{Y}^{B_i}(2k)+\alpha\end{cases} \tag{7}$$

其中，$i=1,2,3,4$ 代表亮度子块序号，$k=1,2,3$ 代表水印嵌入位置，α 代表水印嵌入强度，本文选取 $\alpha=0.5$。

对于含水印的压缩测量值 $\overline{Y}^{B_i}$，采用 OMP 算法对高频系数进行重构。计算重构的 DCT 系数 $\hat{X}^{B_i}=OMP(\overline{Y}^{B_i},\Phi)$，修改原始亮度子块的 DCT 系数向量 $Z^{B_1}\in R^{64}$ 的后 32 个值，由向量 $\hat{X}^{B_i}$ 进行替换，采用逆 DCT 变换恢复第 i 个亮度子块的亮度值。

Step7 按 Step3—Step6 处理所有 I 帧的每个宏块的亮度子块，完成认证水印的生成嵌入过程。

4.2 视频 I 帧的帧内篡改检测

利用认证水印进行帧内篡改检测时，对每个宏块进行二次压缩感知测量后，部分 DCT 系数在进行帧内篡改检测时会出现一定的误判。为了降低误判率，本文利用 I 帧篡改块相邻特性，采用统计阈值做进一步判断，以去除大部分的误判块。

在视频的解码端，I 帧的认证水印提取与帧内篡改检测

算法过程如下。

Step1　按 4.1 节 Step1－Step4，结合密钥压缩测量矩阵 Φ，抽取宏块 MB_{ij} 内的压缩测量向量：$Y^{B_i}{}'$ 和 $\overline{Y}^{B_i}{}'$（$i=1,2,3,4$）。以 $Y^{B_i}{}'$ 为依据，按照式(5)计算宏块内各个子块对应的水印值：$w_i{}'\in\{0,1\}$，$i=1,2,3,4$。

Step2　根据高频系数压缩测量向量 $\overline{Y}^{B_i}{}'$，分 3 次提取嵌入的内容认证水印：$w''(i,k)\in\{0,1\}$，$i=1,2,3,4;k=1,2,3$。提取公式如下：

$$w''(i,k)=\begin{cases}0, & \text{if } (\overline{Y}^{B_i}{}'(2k)<\overline{Y}^{B_i}{}'(2k-1))\\ 1, & \text{if } (\overline{Y}^{B_i}{}'(2k)\geqslant\overline{Y}^{B_i}{}'(2k-1))\end{cases}\tag{8}$$

采用多数投票函数 $Maj(\cdot)$，形成每个亮度子块的水印值：$\overline{w}_i{}'\in\{0,1\}$，$i=1,2,3,4$。提取公式如下：

$$\overline{w}_i{}'=Maj(w''(i,1),w''(i,2),w''(i,3))\tag{9}$$

Step3　对每个视频 I 帧，生成一个初值全为 0 的篡改记录矩阵 TR，用于记录 I 帧内宏块对应全部子块的篡改状态，其大小为 $(N_1/8)\times(N_2/8)$。设子块 B_i 对应的矩阵元素为 $TR(l,h)$，按照式(10)对各个子块进行篡改初检：

$$TR(l,h)=\begin{cases}1, & \text{if}(w_i{}'\neq\overline{w}_i{}')\\ 0, & \text{if}(w_i{}'==\overline{w}_i{}')\end{cases}\tag{10}$$

Step4　误判的修正过程。为了降低误判率，根据篡改记录矩阵 TR，计算 I 帧内任意子块 B_i 的邻域内的篡改相关系数 $TC(l,h)$：

$$TC(l,h)=\sum_{\substack{1\leqslant i,j\leqslant N_1/8\\ |i-l|\leqslant d,|j-h|\leqslant d}}TR(i,j)/d^2\tag{11}$$

如果篡改相关系数 $TC(l,h)>T_C$（阈值），则标注子块 B_i 被篡改。本文选取 $d=2$，$T_C=0.3$。

Step5　按 Step1－Step4 处理宏块中的所有子块，最后在 I 帧图像中显示内容认证结果。

Step6　重复 Step1－Step5，完成视频中所有 I 帧的帧内篡改检测。

5　视频完整性水印

5.1　视频完整性水印生成与嵌入

完整性水印主要用于检测在时间维度上对视频进行的帧删除、帧插入、帧交换等攻击。通过对 I 帧、P 帧的序列值的二值运算生成完整性水印，按照嵌入位置矩阵，分多次嵌入到 I 帧宏块的 C1 色度块的压缩测量值中或者 P 帧亮度子块的压缩测量值中。嵌入与提取流程如图 3 所示。

半脆弱的完整性水印的生成和嵌入算法过程如下。

Step1　对视频 I 帧和 P 帧按照 GOP 次序进行混合编码。设 $F_i(i=0,1,\cdots,N_F-1)$ 表示第 i 帧的序列号（$0\leqslant F_i\leqslant 2^H-1$），用二进制编码表示如下：

$$F_i=(b_{H-1}\cdots b_1b_0)\tag{12}$$

把二进制位 $\hat{w}_i=b_i(i=0,1,\cdots,H-1)$ 作为完整性水印，分多次嵌入到相应的宏块中。本文选取 $H=8$，重复嵌入次数为 3。当视频帧大小为 256×256 时，每帧包含的宏块数量为 256，嵌入 8 位完整性水印至少需要 24 个宏块，能完全满足完整性水印 $\hat{w}_i$ 的嵌入要求。

Step2　选取矩阵 $TL=(TL_{ij})_{H\times 3}$ 作为完整性水印的嵌入位置矩阵，水印 $\hat{w}_i$ 被重复嵌入到编号为 TL_{i0}、TL_{i1}、TL_{i2} 的宏块中。为了提高水印的抗攻击能力，采用混沌 logistic 序列函数 $x_{n+1}=\mu x_n(1-x_n)$，$x_0=K_1$ 计算 TL_{ij} 的值，选取 $3.5699456<\mu\leqslant 4$ 以及 K_1 作为完整性水印生成与提取的密钥。本文取 $\mu=3.5699458$，$K_1=0.35$。

Step3　若当前帧是 I 帧，为了不影响 I 帧的内容认证水印，对于第 i 个完整性水印 $\hat{w}_i$，按照式(6)、式(7)把水印嵌入到 I 帧中编号为 TL_{i0}、TL_{i1}、TL_{i2} 的 3 个宏块的 C1 色度子块中。

Step4　若当前帧是 P 帧，对于第 i 个完整性水印 $\hat{w}_i$，按照式(6)、式(7)把水印 $\hat{w}_i$ 嵌入到 P 帧中编号为 TL_{i0}，TL_{i1}，TL_{i2} 的 3 个宏块的 B1 亮度子块中。

Step5　重复 Step1－Step4，直至完成所有视频帧的完整性水印的嵌入。

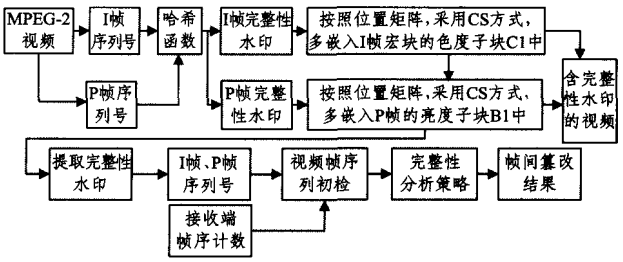


图 3　完整性水印嵌入与提取流程

5.2　视频帧间篡改检测

在视频的解码端，视频完整性水印提取与帧间篡改检测算法过程如下。

Setp1　根据完整性水印密钥 μ、K_1，获取当前帧第 i 个完整性水印的嵌入宏块位置 TL_{i0}，TL_{i1}，TL_{i2}。若当前帧是 I 帧，按照式(8)、式(9)在宏块的色度子块 C1 中分 3 次提取完整性水印，并采用多数投票策略获得视频 I 帧的 $\hat{w}_i{}'$。若当前帧是 P 帧，按照式(8)、式(9)在宏块的亮度子块 B1 中分 3 次提取完整性水印，并采用多数投票策略获得视频 P 帧的 $\hat{w}_i{}'$。

Setp2　根据完整性水印 $\hat{w}_i{}'$，计算当前帧的帧序列值：

$$F_i{}'=\sum_{j=0}^{H-1}(\hat{w}_j{}'\times 2^j)\tag{13}$$

Setp3　接收帧序列值集合为 $F_i{}'(i=0,1,\cdots,N_F{}'-1)$，$N_F{}'$ 是接收到的视频帧总数。结合如下视频完整性分析策略，对视频帧间篡改进行检测。

1）若 $F'_{i_0}=0(i_0>0)$ 且 $\#\{l|F_l{}'=0,|l-i_0|\leqslant\Delta\}\geqslant\tau$，表明从视频的第 i_0 帧开始发生帧插入篡改攻击，Δ 代表当前视频帧的邻域，τ 是错误帧数的阈值，本文取 $\Delta=5$，$\tau=3$。

2）若 $F'_{i_0}\neq F_{i_0}(i_0\geqslant 0)$ 且 $\#\{l|F_l{}'-F_l=F'_{i_0}-F_{i_0},|l-i_0|\leqslant\Delta\}\geqslant\tau$，表明从视频的第 i_0 帧开始发生帧删除篡改攻击。

3）若 $(F'_{i_0}=F_{j_0})\&\&(F'_{j_0}=F_{i_0})(i_0,j_0\geqslant 0)$ 且 $\{F_l{}'|\text{if}(|l-i_0|\leqslant\Delta)\}=\{F_h{}'|\text{if}(|h-j_0|\leqslant\Delta)\}$，表明视频的第 i_0 帧和第 j_0 帧处发生帧交换篡改攻击。

Setp4　重复 Step1－Step3，输出帧间篡改检测结果。

6　实验结果与分析

为了验证本文提出的基于 CS 的视频双水印篡改检测算法，采用标准的 CIF 视频序列进行仿真实验，其中包含 6 个标准的视频片段，视频的基本参数如表 1 所列。

表 1　实验视频序列参数

视频序列	Football	Foreman	Tennis	Container	Monitor	Coastguard
视频大小	352×288	352×288	352×240	352×240	352×240	352×240
I/P 帧间距	5	5	5	5	5	5
色差格式	4:2:0	4:2:0	4:2:0	4:2:0	4:2:0	4:2:0
帧数	125	90	110	90	90	60
帧速(帧/秒)	30	24	24	25	25	25

本文的仿真实验环境：开发平台为 Matlab2012 和 Visual Studio 2010，测试平台为联想 M440S PC，硬件配置为 CPU：Intel(R) I5-4200U4×2.4GHz，内存：4GB DDR3L，操作系统为 32 位的 Windows7 SP1。

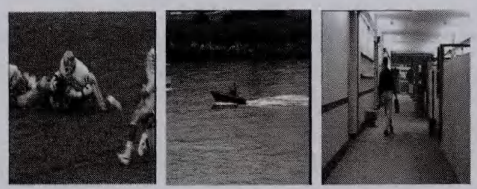
6.1　视频嵌入水印效果分析

为验证本文算法的有效性，通过仿真实验检验嵌入水印后视频的视觉效果。原始视频系列如图 4 所示，嵌入水印后的视频序列如图 5 所示，从中可以看出，嵌入水印后的视频在视觉上几乎没有区别，水印透明性很好。嵌入 CS 水印前后视频的平均 PSNR(峰值信噪比)计算结果如表 2 所列，其 PSNR 都高于 33dB，表明该水印具有良好的不可感知性。

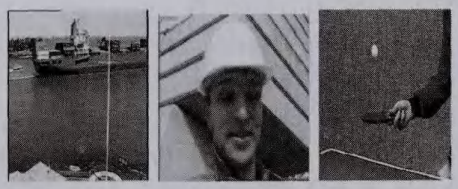
(a)Football　(b)Coastguard　(c)Monitor

(d)Container　(e)Foreman　(f)Tennis

图 4　原始视频系列



(a)Football　(b)Coastguard　(c)Monitor



(d)Container　(e)Foreman　(f)Tennis

图 5　嵌入水印后的视频系列

表 2　视频嵌入 CS 水印前后的测试结果(dB)

视频序列	Football	Coastguard	Monitor	Container	Foreman	Tennis
平均 PSNR	35.06	41.72	43.75	40.01	42.76	33.92

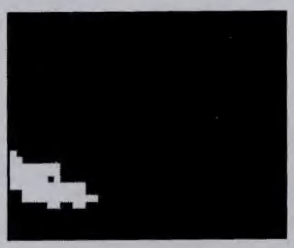
6.2　视频帧内篡改检测分析

视频帧内篡改又称为空间域篡改。本文以 Coastguard、Container、Monitor 视频序列为测试对象，对嵌入水印后的视频序列进行帧图像插入、帧图像修改和帧图像删除的帧内篡改操作。

1)图像插入篡改

在 Coastguard 视频的第 6 帧图像的左下角插入“鱼跃水面”画面，含水印的视频帧如图 6(a)所示，图像插入篡改后的视频帧如图 6(b)所示，对图像插入篡改的定位结果如图 6(c)所示。从图中可以看出，插入的篡改能被精确地检测出来。

(a)含水印的视频　(b)图像插入篡改后的视频

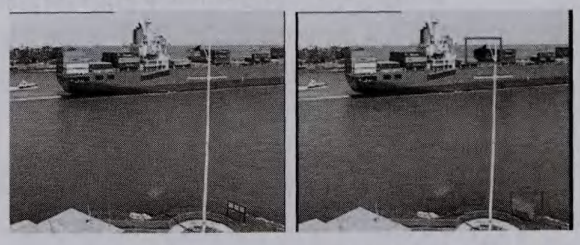


(c)对图像插入篡改的定位结果

图 6　Coastguard 视频的图像插入篡改及检测结果

2)图像修改篡改

在 Container 视频的第 11 帧图像上进行了两处修改，含水印的视频帧如图 7(a)所示，图像修改篡改后的视频帧如图 7(b)所示，对图像修改篡改的定位结果如图 7(c)所示。从图中可以看出，修改的篡改能被清楚地检测出来。



(a)含水印的视频　(b)图像修改篡改后的视频

(c)对图像修改篡改的定位结果

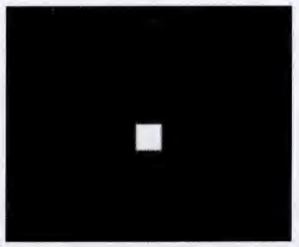
图 7　Container 视频的图像修改篡改及检测结果

3)图像删除篡改

在 Monitor 视频的第 61 帧图像上删除“公文包”，含水印的视频帧如图 8(a)所示，图像删除篡改后的视频帧如图 8(b)所示，对图像删除篡改的定位结果如图 8(c)所示。从图中可以看出，删除的篡改能被精确地检测出来。

(a)含水印的视频

(b)图像删除篡改后的视频



(c)对图像删除篡改的定位结果

图 8　Monitor 视频的图像删除篡改及检测结果

为了验证视频帧内篡改检测的整体准确度，定义帧内篡改检测率计算公式如式(14)所示：

$$R_B=\frac{B_C}{B_T} \tag{14}$$

其中，B_C 为检测出的篡改块数，B_T 为篡改的总块数。

采用 3 段视频为样本，表 3 给出了本文算法和文献[15]算法的篡改检测块数和篡改检测率。对比结果表明，本文算法比文献[15]算法的篡改检测率有明显提高。

表 3　帧内篡改检测率分析

视频序列	篡改区域	本文算法		文献[15]算法	
		检测块数	篡改检测率	检测块数	篡改检测率
Coastguard	30	29	0.97	27	0.9
Container	30	30	1	28	0.933
Monitor	35	34	0.97	32	0.914

6.3　视频帧间篡改检测分析

对 Container 和 Foreman 视频加入水印之后，按表 4 的方式进行帧间篡改。图 9 是篡改后的视频序列的展示，表 5 列出对上述帧间篡改的检测结果。

为了验证视频帧间篡改检测能力，定义帧间篡改检测率的计算公式如式(15)所示：

$$R_F=\frac{F_C}{F_T} \tag{15}$$

其中，F_C 为检测出的视频帧数，F_T 为篡改的视频总帧数。

表 6 给出了本文算法和文献[14]算法对 Container 和 Foreman 视频序列的帧间篡改检测帧数和篡改检测率。从表 6中可以看出，本文算法比文献[14]算法的帧间篡改检测能力更高。

表 4　帧间篡改方式

视频	篡改方式	篡改描述
Container	帧交换	交换第 30 帧和第 94 帧
	帧删除	删除第 46 至第 57 帧
	帧插入	在第 87 帧后插入 3 帧
Foreman	帧交换	交换第 55 帧和第 87 帧
	帧删除	删除第 73 至第 84 帧
	帧插入	在第 84 帧后插入 4 帧

(a)原始 Foreman 视频

(b)交换第 55 帧和第 87 帧之后的 Foreman 视频

(c)删除第 73 至第 84 帧之后的 Foreman 视频

(d)在第 84 帧后插入 4 帧之后的 Foreman 视频

图 9　帧间篡改后的视频序列

表 5　帧间篡改检测结果

视频	篡改方式	篡改说明	篡改检测结果
Container	帧交换	交换第 30 帧和第 94 帧	第 30 帧是错误的，第 94 帧是错误的，
	帧删除	删除第 46 至第 57 帧	有 12 帧是属于帧删除篡改，帧序号是第 46 至第 57 帧
	帧插入	在第 87 帧后插入 3 帧	有 3 帧是属于帧插入篡改，帧序号是第 88 至第 90 帧
Foreman	帧交换	交换第 55 帧和第 87 帧	第 55 帧有可能出错，第 87 帧是错误的
	帧删除	删除第 73 至第 84 帧	有 12 帧是属于帧删除篡改，帧序号是第 73 至第 84 帧
	帧插入	在第 84 帧后插入 4 帧	有 4 帧是属于帧插入篡改，帧序号是第 85 至第 88 帧

表 6　视频帧间篡改检测率对比

视频序列	篡改类型	篡改帧数	本文算法		文献[14]算法	
			检测帧数	篡改检测率	检测帧数	篡改检测率
Container	帧交换	10	10	1	10	1
	帧删除	8	8	1	7	0.875
	帧插入	10	9	0.90	9	0.90
Foreman	帧交换	10	10	1	10	1
	帧删除	12	12	1	11	0.916
	帧插入	15	14	0.933	13	0.867

结束语　本文提出一种基于压缩感知的双水印算法用于实现对视频的帧内篡改与帧间篡改的检测。基于 CS 测量值的安全性与计算保密性，在帧内子块的 DCT 高频系数的压缩测量中嵌入帧内容认证水印与帧序列完整性水印。算法把 CS 理论应用于内容特征提取、水印生成、水印嵌入及提取的全过程中。仿真实验表明，所提算法的篡改检测效果良好，为 CS 理论应用于视频分析中做了新的探索。

参 考 文 献

[1] Su P C, Wu C S, Chen I F, et al. A practical design of digital video watermarking in H. 264/AVC for content authentication [J]. Signal Processing: Image Communication, 2011, 26(8/9): 413-426

[2] Zhang Wei-wei, Zhang Ru, Liu Xian-yi. A video watermarking algorithm of H. 264/AVC for content authentication [J]. Journal of Networks, 2012, 7(8): 1150-1154

[3] Qin Yun-long, Sun Guang-ling, Wang Shuo-zhong, et al. Blind detection of video sequence montage based on GOP abnormality [J]. Acta Electronica Sinica, 2010, 38(7): 1597-1602(in Chinese)
秦运龙，孙广玲，王朔中，等. 根据 GOP 异常进行视频序列剪辑篡改的盲检测[J]. 电子学报，2010，38(7)：1597-1602

[4] Lin You-ru, Huang Hui-yu, Hsu W. An embedded watermark technique in video for copyright protection [C]// Proc of the 18th Int Conf on Pattern Recognition(ICPR). Piscataway, NJ: Institute of Electrical and Electronics Engineers Inc, 2007

[5] De Roover C, De Vleeachouwer C, et al. Robust video hashing based on radial projections of key frames [J]. IEEE Trans on Signal Processing, 2008, 53(10): 4020-4037

[6] Zheng Zhen-dong, Wang Pei, Li Li. A hybrid video watermarking scheme based on motion vectors and DCT [J]. Journal of Image and Graphics, 2009, 14(12): 2631-2634(in Chinese)
郑振东，王沛，李莉. 基于运动矢量域与 DCT 域的混合视频水印方案[J]. 中国图象图形学报，2009，14(12)：2631-2634

[7] Park J Y, Lin J H, Kin G S, et al. Invertible semi fragile watermarking algorithm distinguishing MPEG-2 compression from malicious manipulation[C]// Proceedings of International Conference on Consumer Electronics. New York, NY, USA: IEEE Press, 2002: 18-19

[8] Zeng Xiao, Chen Zhen-yong, Fan Wei, et al. Inverrtibal semi-fragile video watermarking algorithm used for content authentication [J]. Journal of Image and Graphics, 2010, 15(8): 1189-1195(in Chinese)
曾骁，陈真勇，范围，等. 用于内容认证的半脆弱可逆视频水印算法[J]. 中国图象图形学报，2010，15(8)：1189-1195

[9] Ling Chen, Ur-Rehman O, Zhang Wen-jun. Semi-fragile watermarking scheme for H. 264/AVC video content authentication based on manifold feature [J]. KSII Transactions on Internet and Information Systems, 2014, 8(12): 4568-4587

[10] Xu Da-wen, Wang Rang-ding, Wang Ji-cheng. A novel watermarking scheme for H. 264/AVC video authentication [J]. Signal Processing: Image Communication, 2011, 26(6): 267-279

[11] Shi Yan-jiao, Qi Miao, Yi Yu-gen, et al. Object based dual watermarking for video authentication [J]. Optik, 2013, 124(19): 3827-3834

[12] Zhou Yan, Zhang De-feng, Ma Zi-long. The research of image hash watermarking algorithm based on compressed sensing [J]. Journal of Sun Yatsen University (Natural Science Edition), 2010, 49(6): 58-63(in Chinese)
周燕，张德丰，马子龙. 基于压缩传感的图像哈希水印算法研究[J]. 中山大学学报(自然科学版)，2010，49(6)：58-63

[13] Rachbin Y, Baron D. The secrecy of compressed sensing measurements[C]// Proc. Forty-sixth Annual Allerton Conference. 2008: 813-817

[14] Zeng Fan-zhi, Lu Yan-sheng, Zhou Yan. The video tamper detection algorithm based on semi-fragile watermark with compressive sensing [J]. Journal of Circuits and Systems, 2011, 16(4): 86-93(in Chinese)
曾凡智，卢炎生，周燕. 基于压缩传感的半脆弱水印的视频篡改检测算法[J]. 电路与系统学报，2011，16(4)：86-93

[15] Zhao Hui-min, Lai Jian-huang, Cai Jun, et al. A video watermarking algorithm for intraframe tampering detection based compressed sensing [J]. Chinese Journal of Electronics, 2013, 41(6): 1153-1158(in Chinese)
赵慧民，赖剑煌，蔡君，等. 一种用于帧内篡改检测的压缩感知视频水印实现算法[J]. 电子学报，2013，41(6)：1153-1158

[16] Xu Zhi-qiang. Compressed sensing [J]. Science China: Math, 2012, 42(9): 865-877(in Chinese)
许志强. 压缩感知[J]. 中国科学：数学，2012，42(9)：865-877

[17] Cai T T, Wang Lie. Orthogonal matching pursuit for sparse signal recovery with noise [J]. IEEE Trans. on Information Theory, 2011, 57(7): 4680-4688

[18] Liu En-tao, Temlyakov V N. The orthogonal super greedy algorithm and applications in compressed sensing [J]. IEEE Trans. on Information Theory, 2012, 58(4): 2040-2047

[19] Do T T, Gan Lu, Nguyen N, et al. Sparsity adaptive matching pursuit algorithm for practical compressed sensing[C]// Proc. of the Fortieth-Second Asilomar Conference on Signals System and Computers. 2008: 581-587

[20] Yang Yi, Liu Bo-wen, Li Yang. DWT-SVD Domain Video Watermarking Algorithm Based on Motion Analysis[J]. Journal of Chongqing University of Technology(Natural Science), 2015, 29(10): 132-138(in Chinese)
阳溢, 刘博文, 李扬. 基于运动分析的 DWT-SVD 域视频水印算法[J]. 重庆理工大学学报(自然科学), 2015, 29(10): 132-138

(上接第 99 页)

于网络样本的恶意行为检出率，由于自编辑恶意样本的恶意行为的触发方式比较单一，且对恶意行为的隐藏能力较差，因此本文方法能够较好地触发恶意行为，具有较好的检测效果。网络搜集的样本的检测率相对较低，这是因为网络搜集的样本的恶意行为触发方式更加多样，增加了恶意代码检测的难度。

表 5 DroidRunner 恶意行为检测率

检测效果 / 样本	识别数量	漏报数量	检测率(%)
网络样本	35	4	88.6
自编辑样本	5	0	100
总计	40	4	89

同时，对 40 个恶意样本使用 6 折交叉验证方法，通过使用 DroidRunner 和不使用 DroidRunner 来检测 Android 恶意程序，实验结果如下。

表 6 为使用 MonkeyRunner 和 DroidRunner 的 6 次交叉验证的实验结果对比，使用 MonkeyRunner 计算 6 次实验检测率的平均值为 84.1%，6 次实验误报率的平均值为16.2%，6 次实验漏报率的平均值为 15.6%。而使用Droid-Runner计算 6 次实验检测率的平均值为 89.6%，6 次实验误报率的平均值为 11.4%，6 次实验漏报率的平均值为9.5%。由实验结果可以看出，使用 DroidRunner 在检测率、误报率和漏报率方面都有较好的表现，平均检测率提高了5.5%，平均误报率下降了 4.8%，平均漏报率下降了 6.1%。

表 6 恶意行为检测结果比较

检测方式 / 检测结果	Monkeyrunner[5-9]	DroidRunner (本文方法)
平均检测率(%)	84.1	89.6
平均误报率(%)	16.2	11.4
平均漏报率(%)	15.6	9.5

通过对以上测试的统计数据分析，本文方法与传统的行为触发方法相比，能够更好地覆盖应用程序运行期间绝大部分的函数调用，对应用程序中的敏感行为触发效果显著，能够较好地检测 Android 应用的恶意行为。

结束语 本文深入研究了 Andorid 恶意代码动态检测过程中的行为触发方法，设计了 Android 恶意行为检测框架，提出了组合事件行为触发模型——DroidRunner，提出了多组合均衡遍历算法，利用组合操作控件等组合事件触发恶意行为，提高了触发恶意行为的概率。在行为监控环节提出了多层行为监控技术，实现了对 Java 代码和本地代码的函数层调用监控，进而实现了对应用程序恶意行为的高效触发。实验证明，本文提出的恶意行为检测方法具有较高的准确性和覆盖率。

参 考 文 献

[1] Hu Wen-jun, Zhao Shuang, Tao Jing, et al. A Detection Method and System Implementation for Android Malware[J]. Journal of Xi'an Jiaotong University, 2013, 47(10): 37-43(in Chinese)
胡文君, 赵双, 陶敬, 等. 一种针对 Android 平台恶意代码的检测方法及系统实现[J]. 西安交通大学学报, 2013, 47(10): 37-43

[2] Cai Zhi-biao, Peng Xin-guang. Detection of Android malware based on system calls[J]. Computer Engineering and Design, 2013, 34(11): 3757-3761(in Chinese)
蔡志标, 彭新光. 基于系统调用的 Android 恶意软件检测[J]. 计算机工程与设计, 2013, 34(11): 3757-3761

[3] Blasing T, Batyuk L. An android application sandbox system for suspicious software detection[C]// Proceedings of the 5th International Conference on Malicious and Unwanted Software. 2010: 55-62

[4] Hao Peng, Sarma C G B. Using probabilistic generative models for ranking risks of Android apps[C]// Proceedings of the 2012 ACM Conference on Computer and Communications Security. 2012: 241-252

[5] Lu Cheng, Yang Yi-xian. Design and Implementation of Malwares Detection System on Android[D]. Beijing: Beijing University of Posts and Telecommunications, 2012(in Chinese)
路程, 杨义先. Android 平台恶意软件检测系统的设计与实现[D]. 北京: 北京邮电大学, 2012

[6] Android developers. Monkeyrunner[OL]. http://developer.android.com/tools/help/ Monkeyrunner_concepts.html

[7] Yang Huan, Zhang Yu-qing, Hu Yu-pu, et al. A Malware Behavior Detection System of Android Applications Based on Multi-Class Features[J]. Chinese Journal of Computers, 2014, 37(1): 15-27(in Chinese)
杨欢, 张玉清, 胡予濮, 等. 基于多类特征的 Android 应用恶意行为检测系统[J]. 计算机学报, 2014, 37(1): 15-27

[8] Spreitzenbarth M, Freiling F, Echtler F, et al. Mobile-sandbox: having a deeper look into android applications[C]// Proceedings of the 28th Annual ACM Symposium on Applied Computing. ACM, 2013: 1808-1815

[9] HierarchyView[OL]. http://developer.android.com/tools/help/hierarchy-viewer.html

[10] Karami M, Elsabagh M, Najafiborazjani P, et al. Behavioral Analysis of Android Applications Using Automated Instrumentation[C]// Proceedings of the International Conference on Software Security and Reliability Companion. 2013: 182-187