Watermark_algorithms_and_his_stability_Leviev_D_A.

Описание алгоритмов встраивания ЦВЗ. (анализ алгоритмов представлен Левиевым Д. А.)

ЦВЗ (цифровой водяной знак).

Описываемые алгоритмы внедряют ЦВЗ в области исходного изображения. Их преимуществом является то, что для внедрения ЦВЗ нет необходимости выполнять вычислительно громоздкие линейные преобразования изображений. ЦВЗ внедряется за счет манипуляций яркостью $l(x,y) \in \{1,...,L\}$ или цветовыми составляющими $(r(x,y), b(x,y), g(x,y))$.

Алгоритм Катера (Kutter).

Пусть изображение имеет RGB-кодировку. Встраивание выполняется в канал синего цвета, так как к синему цвету система человеческого зрения наименее чувствительна. Рассмотрим алгоритм передачи одного бита секретной информации.

Пусть s_i - встраиваемый бит, $I = \{R, G, B\}$ - контейнер, $p = (x, y)$ - псевдослучайная позиция, в которой выполняется вложение. Секретный бит встраивается в канал синего цвета путем модификации яркости $l(p) = 0.299r(p) + 0.587g(p) + 0.114b(p)$:

$$b'(p) = \begin{cases} b(p) + ql(p), & \text{если } s_i = 0, \\ b(p) - ql(p), & \text{если } s_i = 1. \end{cases}, \qquad (*)$$

где q - константа, определяющая энергию встраиваемого сигнала. Ее величина зависит от предназначения схемы. Чем больше q, тем выше робастность вложения, но тем сильнее его заметность.

Извлечение бита получателем осуществляется без наличия у него исходного изображения, то есть вслепую. Для этого выполняется предсказание значения исходного, немодифицированного пиксела на основании значений его соседей. Предлагается для получения оценки пиксела использовать значения нескольких пикселов, расположенных в том же столбце и той же строке. Авторы использовали «крест» пикселов размером 7x7. Оценка $\hat{b}''(p)$ получается в виде

$$\hat{b}''(p) = \frac{1}{4c}\left(-2b''(p)\sum_{i=-c}^{+c} b''(x+i, y) + \sum_{k=-c}^{+c} b''(x, y+k) \right), \qquad (**)$$

где c - число пикселов сверху (снизу, слева, справа) от оцениваемого пиксела $(c = 3)$. Так как в процессе встраивания ЦВЗ каждый бит был повторен cr раз, то мы получим cr оценок одного бита ЦВЗ. Секретный бит находится после усреднения разности оценки пиксела и его реального значения

$$\delta = \frac{1}{cr}\sum_{i=1}^{cr} \hat{b}_i(p) - b_i(p). \qquad (***)$$

Знак этой разности определяет значение встроенного бита.

Можно ли гарантировать всегда верное определение значения секретного бита? Нет, так как функция извлечения бита не является обратной функции встраивания. Для повышения надежности необходимо применение дополнительных мер.

Алгоритм является робастным ко многим из известных атак: низкочастотной фильтрации изображения, его сжатию в соответствии с алгоритмом JPEG, обрезанию краев.

Алгоритм Брундокса (Bruyndonckx).

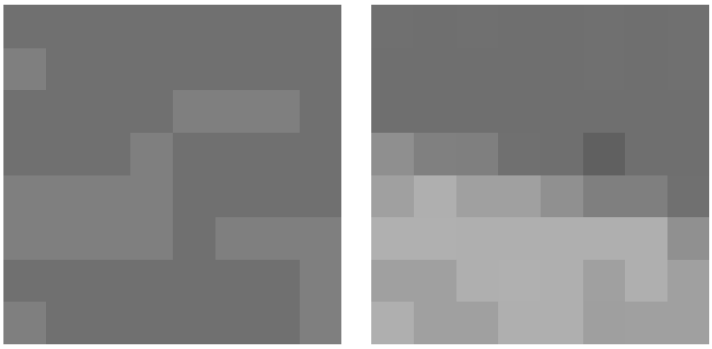
ЦВЗ представляет собой строку бит. Для повышения помехоустойчивости применяется код БЧХ (код исправляющий кратные ошибки). Внедрение осуществляется за счет модификации яркости блока 8x8 пикселов.

Процесс встраивания осуществляется в три этапа.

1) Классификация, или разделение пикселов внутри блока на две группы с примерно однородными яркостями.
2) Разбиение каждой группы на категории, определяемые данной сеткой.
3) Модификация средних значений яркости каждой категории в каждой группе.

Рассмотрим подробнее каждый из этих этапов.

1) При классификации авторы выделяют два типа блоков: блоки с «шумовым контрастом» (а) и блоки с резко выраженными перепадами яркости (б).



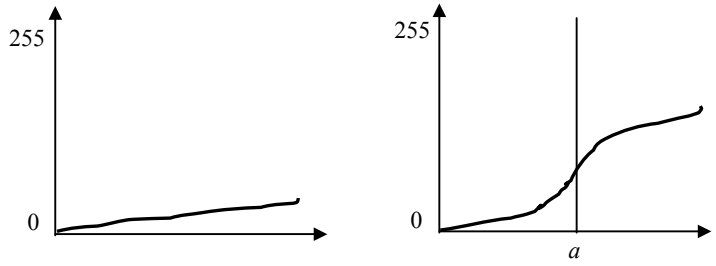
(а) (б)

Два типа блока: а) с нечетким контрастом и
б) с резко выраженным контрастом

В блоках второго типа зоны с отличающейся яркостью не обязательно должны располагаться вплотную друг к другу, не обязательно должны содержать равное количество пикселов. Более того, некоторые пикселы вообще могут не принадлежать ни одной зоне. В блоках первого типа классификация особенно затруднена.

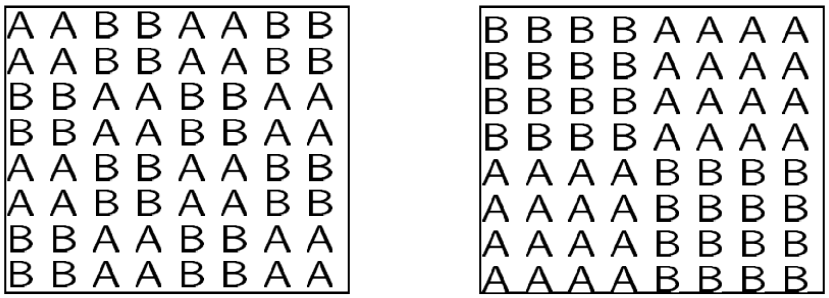
Для выполнения классификации значения яркости сортируются по возрастанию ((а) и (б)). Далее находится точка, в которой наклон касательной к получившейся кривой максимален (α). Эта точка является границей, разделяющей две зоны в том случае, если наклон больше некоторого порога. В противном случае пикселы делятся между зонами поровну.

2) Для сортировки пикселов по категориям на блоки накладываются маски, разные для каждой зоны и каждого блока. Назначение масок состоит в обеспечении секретности внедрения. Пример масок для двух зон приведен на (а) и (б).



(а) (б)

Сортированные значения яркостей блоков



Пример используемых масок

3) Модификация. Итак, множество пикселов оказалось разделенным на пять подмножеств: две зоны * две категории + пикселы, не принадлежащие какой-либо зоне (для блоков первого типа). Обозначим среднее значение яркости для пикселов двух зон и категорий через $l_{1A}, l_{2A}, l_{1B}, l_{2B}$. Нам известно, что $l_{1A} < l_{2A}$, $l_{1B} < l_{2B}$. Встраивание бита ЦВЗ s осуществляется по следующему правилу:

$$s=\begin{cases}1, & \begin{cases}l'_{1A}>l'_{1B},\\ l'_{2A}>l'_{2B},\end{cases}\\ 0, & \begin{cases}l'_{1A}<l'_{1B},\\ l'_{2A}<l'_{2B}.\end{cases}\end{cases} \qquad (*)$$

С другой стороны, необходимо обеспечить равенство значений яркости в каждой зоне:

$$\frac{n_{1A}l'_{1A}+n_{1B}l'_{1B}}{n_{1A}+n_{1B}}=l_1 \text{ и } \frac{n_{2A}l'_{2A}+n_{2B}l'_{2B}}{n_{2A}+n_{2B}}=l_2 . \qquad (**)$$

Для достижения этого яркость всех пикселов одной зоны меняется одинаково. Например, для зоны 1, категории А это изменение составит $l'_{1A}-l_{1A}$.

Алгоритм извлечения ЦВЗ является обратным алгоритму внедрения. При этом вычисляются средние значения яркостей и находятся разности

$$s''=\begin{cases}0, & \text{если } l''_{1A}-l''_{1B}<0 \text{ и } l''_{2A}-l''_{2B}<0\\ 1, & \text{если } l''_{1A}-l''_{1B}>0 \text{ и } l''_{2A}-l''_{2B}>0.\end{cases} \qquad (***)$$

Алгоритм Ленгелаара Langelaar

Данный алгоритм также работает с блоками 8x8. Вначале создается псевдослучайная маска нулей и единиц такого же размера $pat(x,y)\in\{0,1\}$. Далее каждый блок B делится на два субблока B_0 и B_1, в зависимости от значения маски. Для каждого субблока вычисляется среднее значение яркости, l_0 и l_1. Далее выбирается некоторый порог α, и бит ЦВЗ встраивается следующим образом:

$$s=\begin{cases}1, & l_0-l_1>+\alpha,\\ 0, & l_0-l_1<-\alpha.\end{cases} \qquad (*)$$

Если условие (*) не выполняется, мы изменяем значение яркости пикселов субблока B_1. Для извлечения бита ЦВЗ вычисляются средние значения яркости субблоков - l''_0 и l''_1. Разница между ними позволяет определить искомый бит:

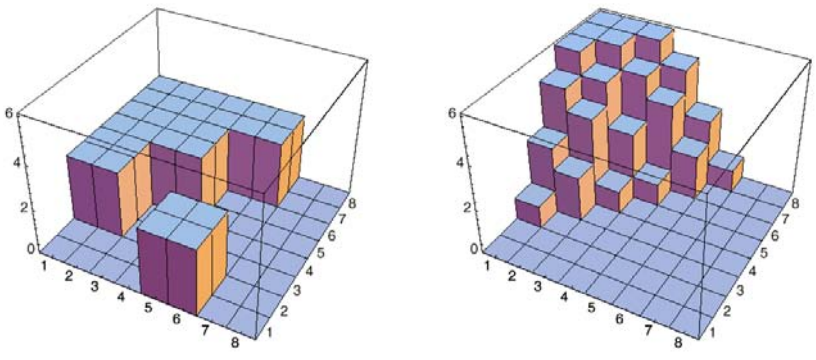
$$s=\begin{cases}1, & l_0-l_1>0,\\ 0, & l_0-l_1<0.\end{cases} \qquad (**)$$

Алгоритм Питаса (Pitas).

ЦВЗ представляет собой двумерный массив бит размером с изображение, причем число единиц в нем равно числу нулей. Существует несколько версий алгоритма, предложенного Питасом. Вначале предлагалось встраивать бит ЦВЗ в каждый пиксел изображения, но позже благоразумно было решено использовать для этой цели блоки размером 2x2 или 3x3 пиксела, что делает алгоритм более робастным к сжатию или фильтрации. ЦВЗ складывается с изображением:

$$l'(x,y)=l(x,y)+\alpha s(x,y). \qquad (*)$$

В случае использования для внедрения блоков детектор ЦВЗ вычисляет среднее значение яркости этого блока. Отсюда появляется возможность неравномерного внедрения ЦВЗ в пикселы, то есть величина $\alpha\neq const$. Таким образом можно получить ЦВЗ, оптимизированный по критерию робастности к процедуре сжатия алгоритмом JPEG. Для этого в блоке 8x8 элементов заранее вычисляют «емкость» каждого пиксела (с учетом ДКП и матрицы квантования JPEG). Затем ЦВЗ внедряют в соответствии с вычисленной емкостью. Эта оптимизация производится раз и навсегда, и найденная маска применяется для любого изображения. На рис. (а) и (б) показан ЦВЗ до и после оптимизации.



Алгоритм Роджена Rongen

. Также, как и в предыдущем алгоритме, ЦВЗ представляет собой двумерную матрицу единиц и нулей с примерно равным их количеством. Пикселы, в которые можно внедрять единицы (то есть робастные к искажениям), определяются на основе некоторой характеристической функции (характеристические пикселы). Эта функция вычисляется локально, на основе анализа соседних пикселов. Характеристические пикселы составляют примерно 1/100 от общего числа, так что не все единицы ЦВЗ встраиваются именно в эти позиции. Для повышения количества характеристических пикселов в случае необходимости предлагается осуществлять небольшое предыскажение изображения.

Детектор находит значения характеристических пикселов и сравнивает с имеющимся у него ЦВЗ. Если в изображении ЦВЗ не содержится, то в характеристических пикселах количество единиц и нулей будет примерно поровну. Авторы рассчитали значение порога принятия решения, минимизирующего вероятность ложной тревоги.

Алгоритм PatchWork.

В основе алгоритма Patchwork лежит статистический подход. Вначале псевдослучайным образом на основе ключа выбираются два пиксела изображения. Затем значение яркости одного из них увеличивается на некоторое значение (от 1 до 5), значение яркости другого – уменьшается на то же значение. Далее этот процесс повторяется большое число раз (~10000) и находится сумма значений всех разностей. По значению этой суммы судят о наличии или отсутствии ЦВЗ в изображении.

Для пояснения работы алгоритма введем ряд обозначений. Пусть значения выбираемых на каждом шаге пикселов a_i и b_i, величина приращения - δ. Тогда сумма разностей значений пикселов

$$S_n = \sum_{i=1}^{n} \left[\left(a_i + \delta \right) - \left(b_i - \delta \right) \right] = 2\delta n + \sum_{i=1}^{n} \left(a_i - b_i \right) \qquad (*)$$

Матожидание величины $\sum_{i=1}^{n} \left(a_i - b_i \right)$ (суммы разности значений пикселов в незаполненном контейнере) близко к нулю при достаточно большом n. Матожидание величины S_n будет больше 2δ. Показано, что S_n имеет гауссовское распределение. Таким образом, в стегодетекторе в соответствии с ключом проверяется значение S_n и в том случае, если она значительно отличается от нуля, выносится решение о наличии ЦВЗ.

Авторами также предложены улучшения основного алгоритма для повышения его робастности. Вместо отдельных пикселов предлагается использовать блоки, или patches. Отсюда и название алгоритма. Использование блоков различного размера может рассматриваться как формирование спектра вносимого ЦВЗ шума (шейпинг), аналогично тому, как это применяется в современных модемах. Так как наиболее вероятной модификацией стего является компрессия JPEG, то целесообразно, чтобы спектр ЦВЗ находился в области низких частот. С другой стороны, если характер возможных модификаций стего заранее неизвестен, целесообразно применение сигналов с расширенным спектром. От формы блока зависит невидимость вносимых искажений.

Алгоритм Patchwork является достаточно стойким к операциям сжатия изображения, его усечения, изменения контрастности. Основным недостатком алгоритма является его неустойчивость к афинным преобразованиям, то есть поворотам, сдвигу, масштабированию. Другой недостаток заключается в малой пропускной способности. Так, в базовой версии алгоритма для передачи 1 бита скрытого сообщения требуется 20000 пикселов.

Алгоритм Бендера Bender.

Это алгоритм основанный на копировании блоков из случайно выбранной текстурной области в другую, имеющую сходные статистические характеристики. Это приводит к появлению в изображении полностью одинаковых блоков. Эти блоки могут быть обнаружены следующим образом:

1. Анализ функции автокорреляции стегоизображения и нахождение ее пиков.
2. Сдвиг изображения в соответствии с этими пиками и вычитание изображения из его сдвинутой копии.
3. Разница в местоположениях копированных блоков должна быть близка к нулю. Поэтому можно выбрать некоторый порог и значения, меньшие этого порога по абсолютной величине, считать искомыми блоками.

Так как копии блоков идентичны, то они изменяются одинаково при преобразованиях всего изображения. Если сделать размер блоков достаточно большим, то алгоритм будет устойчивым по отношению к большинству из негеометрических искажений. В проведенных экспериментах показана робастность алгоритма к фильтрации, сжатию, поворотам изображения.

Основным недостатком алгоритма является исключительная сложность нахождения областей, блоки из которых могут быть заменены без заметного ухудшения качества изображения. Кроме того, в данном алгоритме в качестве контейнера могут использоваться только достаточно текстурные изображения.

Один из предложенных способов для проверки аутентичности изображений получил название метода проверочных сумм. Согласно этому методу отбирались семь старших бит восьми близлежащих пикселов. Получалось 56-битное слово. Выполнив эту операцию для всего изображения, имели $N \times N / 8$ таких слов, где $N \times N$ - число пикселов в изображении. Затем они поразрядно складывались по модулю два, то есть вычислялась проверочная сумма длиной 56 бит. Эта сумма записывалась в младшие значащие биты выбранных в соответствии с ключом пикселов. В детекторе осуществлялась проверка этих бит, получившаяся проверочная сумма сравнивалась с эталонной, и выносилось решение о наличии или отсутствии модификации изображения. Таким образом, в данном алгоритме в качестве ключа использовались местоположение несущих проверочную сумму пикселов и сама эта проверочная сумма.

Рассмотренные алгоритмы легли в основы множества программ так или иначе связанных с ЦВЗ .Популярность мультимедиа-технологий вызвало множество исследований, связанных с разработкой алгоритмов ЦВЗ для использования в стандартах MP3, MPEG, JPEG2000, защиты DVD дисков от копирования. В частности для mpeg файлов как правило применяются легкие алгоритмы на основе модификации яркости. (camtasia watermark или Sundance Software Technologies) .

Литература:

Kutter M, Jordan F, Bosser F. Digital signature of color images using amplitude modulation.

Lengelaar G Real-time Watermarking Techniques for Compressed Video Data.(прилагается zip)

P.Meerval . Digital image Watermarking in the wavelet Transform Domain (прилагается zip)

Nikolaidis N., Pitas I. Robust image watermarking in the spatial domain.

Maes M., Rongen P., van Overveld C. Digital image waermarking by salient point modification practical results.

Bender W., Gruhl D., Morimoto N., Lu A. Techniques for Data Hiding // IBM Systems Journal. 1996. Vol. 35.

Petitcolas F. Weakness of existing watermarking schemes.
http://www.cl.cam.ac.uk/~fapp2/watermarking/image_watermarking .