Distinctive Scale Invariant Image Watermarking using Hermite Splines

Mohan Allam[1] and Adinarayana Salina[2]

[1]Assistant Professor, Dept of IT, Shri Vishnu Engg. College for Women
Bhimavaram, Andhra Pradesh, India. mohanallam@gmail.com
[2]Associate Professor,Dept of IT, Shri Vishnu Engg. College for Women
Bhimavaram, Andhra Pradesh, India. s_adi_2k@yahoo.com

ABSTRACT: This paper presents a robust non blind watermarking scheme. The proposed scheme can embed the watermark in the colour image. This watermarking scheme is based on a sample spline approach by extracting distinctive invariant features from images. It may be used to do reliable matching among different views of an object. This paper also describes an approach for object recognition using these invariant features. The recognition can be performed by matching individual features to a database of the known features of objects using a fast nearest neighbour algorithm. The low frequency components of image blocks are used for data hiding to obtain high robustness against attacks. We take four samples of the approximation coefficients of the image blocks to construct a spline curve in the 2-D space. Here the slope of this spline curve is invariant to the gain factor, which is used for watermarking purpose. We embed the watermarking code by constructing a spline curve according to watermarking bits. We compute the distribution of the slope of the embedding spline curve for designing a maximum likelihood decoder.

1 INTRODUCTION

Digital watermarking embeds information within a digital work as a part of the media. Watermarking techniques falls into three categories of robust, semi fragile and fragile methods according to their specic applications [1]. Robust watermarking mainly serves for identication purposes while the fragile and semi fragile watermarking are usually employed in authentication applications. Since a good watermarking scheme should always be able to deal with some kinds of attacks, studies in the watermarking research area mostly target robust watermarking problems [2].

Image matching is a fundamental aspect of many problems in computer vision, including object or scene recognition, solving for 3D structure from multiple images. This paper describes image features that have many properties that make them suitable for matching differing images of an object or scene. The features are invariant to image scaling and rotation, and partially invariant to change in illumination and 3D camera viewpoint. The cost of extracting these features is minimized by taking a cascade filtering approach, in which the more expensive operations are applied only at locations that pass an initial test. The major stages of computation used to generate the set of image features are Scale-space extrema detection, Keypoint localization, Orientation assignment, Keypoint descriptor. This approach has been named the Scale Invariant Feature Transform (SIFT) [3], as it transforms image data into scale-invariant coordinates relative to local features.

An important aspect of this approach is that it generates large numbers of features that densely cover the image over the full range of scales and locations. A typical image of size 500×500 pixels will give rise to about 2000 stable features .The quantity of features is particularly important for object recognition, where the ability to detect small objects in cluttered backgrounds requires that at least 3 features be correctly matched from each object for reliable identification. For image matching and recognition, SIFT features are first extracted from a set of reference images and stored in a database. A new image is matched by individually comparing each feature from the new image to this previous database and finding candidate matching features based on Euclidean distance of their feature vectors.

In this paper, we proposed a novel gain invariant watermarking scheme based on a sample spline scheme. Embedding the watermark bits into the approximation coefficients of the image blocks makes the algorithm highly robust against noise and compression attacks. Any possible selection of four approximation coefficients, that may be selected using a secret key, constructs a spline curve whose slope at control points is considered for data hiding. We embed the watermark bits by constructing a spline curve with the tangents at control points (slope). In this way, the slope of the spline curve carries the watermark information while the distortion imposed to its constructive samples is minimal. Since our embedding process is linear, it can be denoted by multiplication of specic embedding matrices. To implement the maximum likelihood (ML) detector for data extraction, we should calculate the distribution of the slope of the spline curve. To this aim, we consider the fact that the approximation coefficients of most of the image blocks can be well-modeled by Gaussian distribution [4].

The rest of the paper is organized as follows. In Section II, we describe the model of the system. The watermark embedding and decoding process are introduced in Section III. Section IV analyzes and evaluates the performance of the proposed scheme and Section V concludes the paper.

Received February 25, 2014

2 SYSTEM MODELING

In this section, we first describe an approach for object recognition using a set of image features.

2.1 Scale-space extrema detection:

As described in the introduction, we will detect keypoints using a cascade filtering approach that uses efficient algorithms to identify candidate locations that are then examined in further detail. The first stage of keypoint detection is to identify locations and scales that can be repeatedly assigned under differing views of the same object. Detecting locations that are invariant to scale change of the image can be accomplished by searching for stable features across all possible scales, using a continuous function of scale known as scale space.

It is implemented efficiently by using a difference-of-Gaussian function to identify potential interest points that are invariant to scale and orientation. An efficient approach to construction of $D(x, y, \sigma)$ is shown in Figure 1.[3]

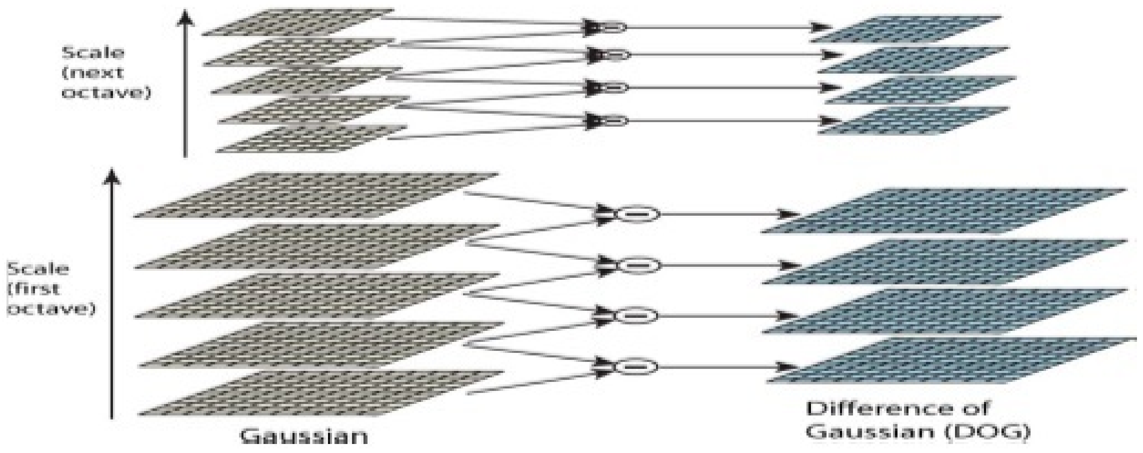


Figure 1: For each octave of scale space, the initial image is repeatedly convolved with Gaussians to produce the set of scale space images shown on the left. Adjacent Gaussian images are subtracted to produce the difference-of-Gaussian images on the right. After each octave, the Gaussian image is down-sampled by a factor of 2, and the process repeated.

2.1.1 Local Extrema Detection

In order to detect the local maxima and minima of $D(x, y, \sigma)$, each sample point is compared to its eight neighbours in the current image and nine neighbours in the scale above and below (see Figure 2). It is selected only if it is larger than all of these neighbours or smaller than all of them. The cost of this check is reasonably low due to the fact that most sample points will be eliminated following the first few checks.

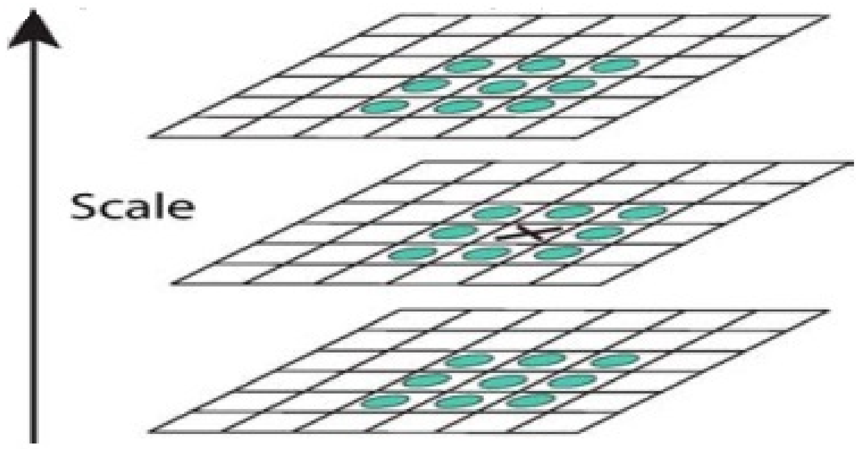


Figure 2: Maxima and minima of the difference-of-Gaussian images are detected by comparing a pixel (marked with X) to its 26 neighbours in 33 regions at the current and adjacent scales (marked with circles).

2.2 Accurate Keypoint Localization:

At each candidate location, a detailed model is fit to determine location and scale. Keypoints are selected based on measures of their stability. Once a keypoint candidate has been found by comparing a pixel to its neighbours, the next step is to perform a detailed fit to the nearby data for location, scale, and ratio of principal curvatures. This information allows points to be rejected that have low contrast (and are therefore sensitive to noise) or are poorly localized along an edge. The initial implementation of this approach simply located keypoints at the location and scale of the central sample point.

2.3 Orientaion Assignment:

One or more orientations are assigned to each keypoint location based on local image gradient directions. All future operations are performed on image data that has been transformed relative to the assigned orientation, scale, and location for each feature, thereby providing invariance to these transformations.

2.4 The Local Image keypoint Descriptor:

The local image gradients are measured at the selected scale in the region around each keypoint. These are transformed into a representation that allows for significant levels of local shape distortion and change in illumination. The previous operations have assigned an image location, scale, and orientation to each keypoint. These parameters impose a repeatable local 2D coordinate system in which to describe the local image region, and therefore provide invariance to these parameters. The next step is to compute a descriptor for the local image region that is highly distinctive yet is as invariant as possible to remaining variations, such as change in illumination or 3D viewpoint.

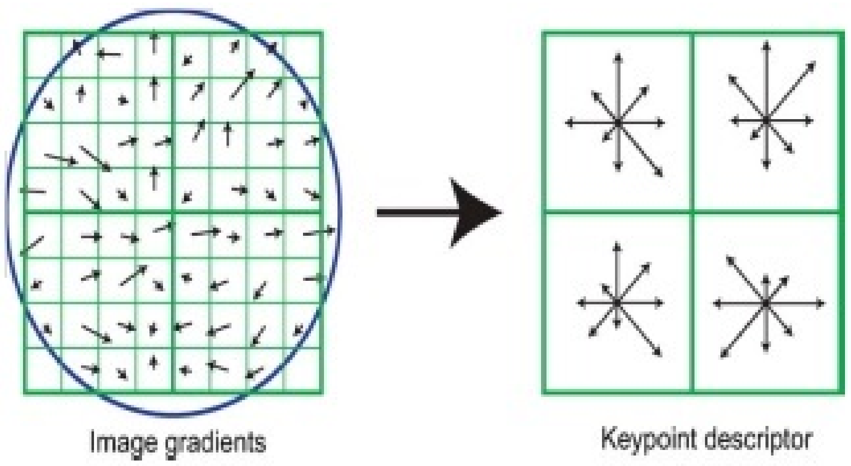


Figure 3: Image gradients and keypoint descriptor.

A keypoint descriptor is created by first computing the gradient magnitude and orientation at each image sample point in a region around the keypoint location, as shown in the Figure 3 [3]. These are weighted by a Gaussian window, indicated by the overlaid circle. These samples are then accumulated into orientation histograms summarizing the contents over 4 X 4 sub regions, as shown on the right, with the length of each arrow corresponding to the sum of the gradient magnitudes near that direction within the region. This figure shows a 2 X 2 descriptor array computed from an 8 x 8 set of samples, whereas the experiments in this paper use 4 X4 descriptors computed from a 16 X 16 sample array.

To this aim, we will detect four samples of an independently and identically distributed (i.i.d) Gaussian random variable from the Local Image keypoint Descriptor. We show this host signal as u = [u1, u2, u3, u4] with the Gaussian distribution of $N(0, \sigma 2u)$. These four samples form two points P1 = [u1, u2] and P2 = [u3, u4] in the 2-D space. We employ Dpk, the slope of the spline at first control point and Dpk+1, the slope of the spline at second control point as our watermarking variables.Figure 4 illustrates these two points as well as the curve derivatives ('slopes') at these points. We can write the procedure as (Algorithm will generate a cubic curve)

Let the parametric curve be $P(u) = au^3 + bu^2 + cu + d$. where u is the parameter that ranges from 0 to 1. A Hermite curve has a defined set of coefficients a, b, c , d. Substituting a value u into the equation gives a point on the Hermite curve. Substituting many values of u from 0 to 1 will trace out the curve. Given the two points and two slopes,$P_0, P_1, P_0', andP_1'$, our objective is to find the coefficients a, b, c, d.

$$P(u) = au^3 + bu^2 + cu + d \tag{1}$$

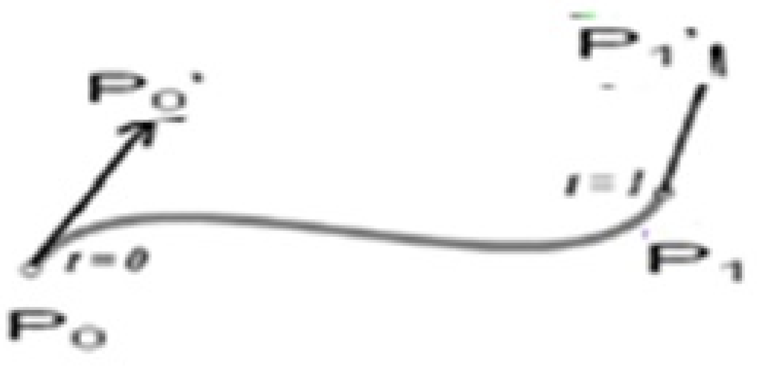


Figure 4: Hermite Specification Control points P_0, P_1 and Slopes P_0', P_1'.

$$P(u) = \begin{pmatrix} u^3 & u^2 & u & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}$$

Derivative of P(u) is,

$$P'(u) = \begin{pmatrix} 3u^2 & 2u & 1 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}$$

But,

$$\begin{aligned} P(0) = p_0 &= a \times 0^3 + b \times 0^2 + c \times 0 + d \times 1 \\ P(1) = p_1 &= a \times 1^3 + b \times 1^2 + c \times 1 + d \times 1 \\ P'(0) = p_0' &= a \times 3 \times 0^2 + b \times 2 \times 0 + c \times 1 + d \times 0 \\ P'(1) = p_1' &= a \times 3 \times 1^2 + b \times 2 \times 1 + c \times 1 + d \times 0 \end{aligned}$$

Therefore, in matrix form,

$$\begin{pmatrix} p_0 \\ p_1 \\ p_0^{'} \\ p_1^{'} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}$$

Solve for a,b,c,d by using matrix inverse:

$$\begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} p_0 \\ p_1 \\ p_0^{'} \\ p_1^{'} \end{pmatrix}$$

We get:

$$\begin{aligned} a &= 2p_0 - 2p_1 + p_0^{'} + p_1^{'} \\ b &= -3p_0 + 3p_1 - 2p_0^{'} - p_1^{'} \\ c &= p_0 \\ d &= p_0 \end{aligned}$$

Next, Substitute back to equation

$$P(u) = au^3 + bu^2 + cu + d$$

We get:

$$P(u) = (2p_0 - 2p_1 + p_0^{'} + p_1^{'})u^3 + (-3p_0 + 3p_1 - 2p_0^{'} - p_1^{'})u^2 + p_0^{'}u$$

Rearranging.

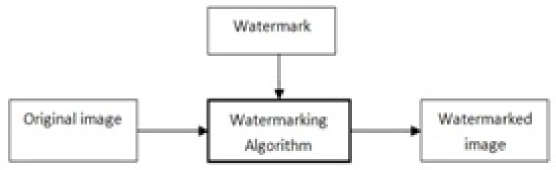
$$P(u) = p_0(2u^3 - 3u^2 + 1) + p_1(-2u^3 + 3u^2) + p_0^{'}(u^3 - 2u^2 + u) + p_1^{'}p_1(u^3 - u)$$

Now, to embed the M-ary watermark code, we use this equation to generate Hermite spline curve given control points and slopes shown in Figure 4, depending on the watermark code. In this way, we obtain the watermarked signal. To extract the hidden bits, an optimum decoder is implemented using M-Hypothesis test as follows. We take the received watermarked signal which consists of the Hermite Spline with two control points and some points on spline curve and calculate the slopes of the Hermite spline curve at each control point. Here the slopes are nothing but the watermarked bits which we have sent it with Host image signal.

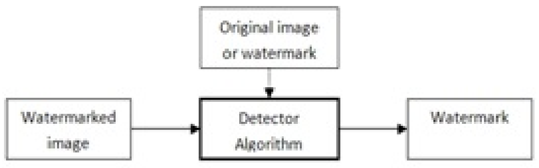
In this paper, a new data hiding method has been proposed. The proposed method takes the received watermarked signal which consists of the Hermite Spline with two control points and some points on spline curve and calculates the slopes of the Hermite spline curve at each control point. Here the slopes are nothing but the watermarked bits which we have sent it with Host image signal.

3 PROPOSED METHOD

In this section, we introduce our blind watermarking scheme. As discussed in the previous section, we assume the host signal as a four-sample i.i.d. Gaussian random signal. In practical applications, these four samples can come from approximation coefficients of the image blocks which satisfy our i.i.d. Gaussian assumption according to Kolmogorov Smirnov test results.



(a) Watermarking Algorithm



(b) Watermark Detection Algorithm

Figure 5: Watermarking and Detection Algorithms

Figure 5a shows a model of watermarking Algorithm in which the watermark is embedded into the original image and finally gets Watermarked image.Figure 5b shows Watermark Detection Algorithm which the Detector Algorithm recovers the Watermark bits from the Watermarked image.

3.1 Watermark embedding:

We use Hermite cubic interpolation for embedding watermark with the help of four samples as control points and Watermark embedding bits as Slopes at each control point. The interpolator used to construct the curve generally has a parametric representation, i.e. the curve is a two dimensional function of an underlying parameters s,

$$x = f(s)$$

Which passes through the collection of sample points $\{ X_i \}$,

$$x_i = f(s_i)$$

This notation allows for easy extension to three dimensions [5]. The parametric notation also gives an interpolator that is isotropic, i.e. invariant with respect to orientation. The parameter s is usually related to the arc length along the curve. However, using the chord length

$$s_{i+1} = s_i + d_i$$

Where

$$d_i = |d_i| = |x_{i+1} - x_i|$$

With s varying linearly between control points, will give equally good results [6]. A further simplification,$s_i = i$, can be made if the point spacing is fairly uniform. The interpolation requirement precludes the use of some functions such as quadratic B-splines [7] which are otherwise suitable curve generators. One simple class of functions that both interpolates and can be made sufficiently smooth are the cubic splines and sub-splines. These functions are piecewise cubic, i.e. on any interval $(s_i, s_(i+1))$ the function is a cubic

$$f_i(s) = a_{3i}s^3 + a_{2i}s^2 + a_{1i}s + a_{0i}$$

For sufficient smoothness, we require that the curve be first derivative (C 1) continuous. Such curves are variously known as sub-splines or Hermite cubics. They are completely defined by the control points { x_i } and the curve derivatives ('slopes') at these points { $\overline{x}_i$ } (Figure 6). Eachcubic segment can also be represented as a linear combination of four basis or 'blending' [8] functions weighted by the end points and the end-point derivatives

$$f(s_i) = x_i\phi_0(t) + \overline{x}_i\phi_1(t) - \overline{x}_(i+1)\phi_1(1-t) + x_{i+1}\phi_1(1-t), t = \frac{s - s_i}{d_i} \in [0, 1]$$

These basis functions, the Hermite Cubic basis functions, are

$$\phi_0(t) = 2t^3 - 3t^2 + 1$$

$$\phi_1(t) = t(1-t)^2$$

And are shown in Figure 6.

Since the sample points { x_i } are given; only the slopes $\overline{x}_i$ need to be determined in order to specify completely the interpolator. The slope values $\overline{x}_i$ are the true derivative values of the interpolator f(s) , but can only be an estimate of the derivatives of the original curve. The overall Hermite cubic interpolate function f(s) is thus composed of piecewise cubic segments h(s) with first-derivative continuity across interval boundaries.

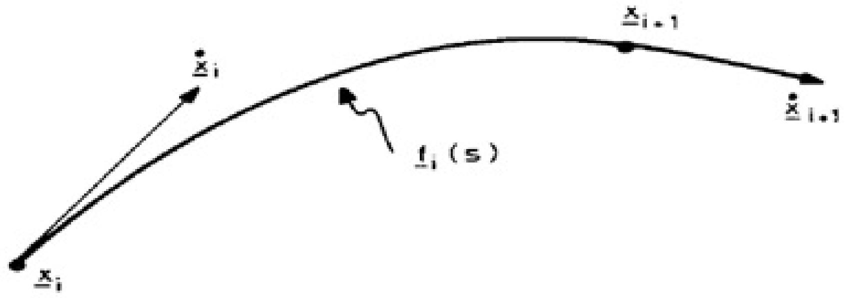
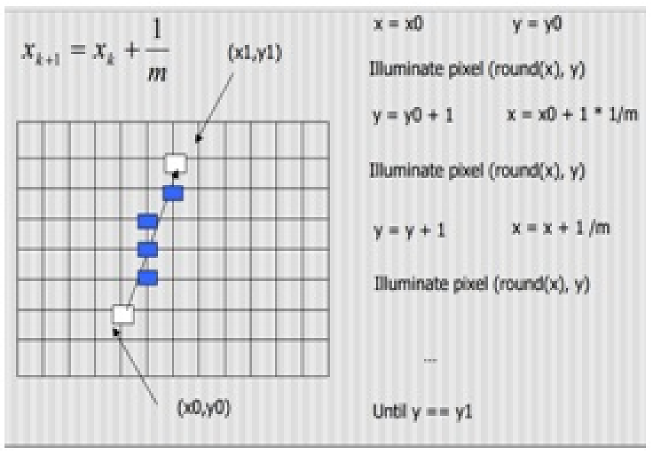
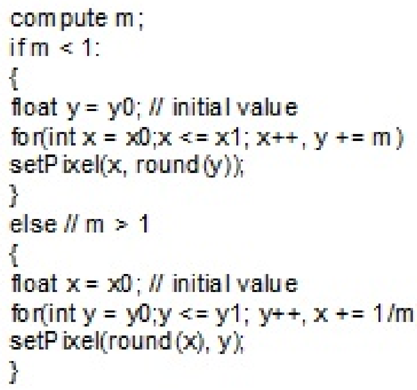


Figure 6: Cubic segment defined by its end points and end-point derivatives.

After finding the points on the spline curve, we embed these points in the pixel positions of a line segment (line joining control points) using A Fibonacci LSB Data Hiding Technique. We can find pixels along a line segment by joining the two control points with the help of digital differential analyzer (DDA) Line Drawing Algorithm. DDA is used for linear interpolation of variables over an interval between start and end point. DDAs are used for rasterization of lines, triangles and polygons. In its simplest implementation the DDA Line drawing algorithm interpolates values in interval $[(x_{start}, y_{start}), (x_{end}, y_{end})]$ by computing for each x_i the equations $x_i = x_{i-1} + \frac{1}{m}$ and $y_i = y_{i-1} + m$ where $\Delta x = x_{end} - x_{start}$, $\Delta y = y_{end} - y_{start}$ and $m = \frac{\Delta y}{\Delta x}$ Figure 7 shows the Pseudocode for the implementation of DDA Line Drawing Algorithm.



```
x = x0                y = y0
Illuminate pixel (round(x), y)
y = y0 + 1            x = x0 + 1 * 1/m
Illuminate pixel (round(x), y)
y = y + 1             x = x + 1 /m
Illuminate pixel (round(x), y)
...
Until y == y1
```



```
compute m;
if m < 1:
{
float y = y0; // initial value
for(int x = x0;x <= x1; x++, y += m)
setPixel(x, round(y));
}
else // m > 1
{
float x = x0; // initial value
for(int y = y0;y <= y1; y++, x += 1/m)
setPixel(round(x), y);
}
```

Figure 7: Line Drawing Algorithm Pseudo code.

3.1.1 Insertion Algorithm

A Fibonacci LSB is a generalization of the classical Least Significant Bit (LSB) and is one of the simplest technique in digital watermarking is in spatial domain using the two dimensional array of pixels in the container image to hold hidden data. The human eyes are not very attuned to small variance in color and therefore processing of small difference in the LSB will not noticeable. A Fibonacci LSB embedding scheme is as follows,

Let us define I (x, y) the cover image, and w (x, y) the data to be embedded. In our experiment they have the same size. The proposed embedding scheme can be summarized as follows:

1. The cover image I (x, y) is decomposed into bit planes I p (x, y) = Fp { I (x, y)} by using the Fibonacci p-decomposition computed using the specified p-sequence

2. The selected plane is considered. For each bit the fulfilment of Zeckendorf condition is checked. If it is verified, the mark is inserted otherwise the following is considered. The simplest method used in our experiment is the substitution of the selected bit value with the corresponding watermark bit. An improved version is based on the following additive scheme:

$Ip(x, y) = Ip(x, y) + \alpha w(x, y)$

3. Once the whole mark has been inserted, the image is reconstructed from the bit planes. The watermarked image is therefore recomposed With respect to this basic scheme; several modifications have been tested to increase the robustness of the system. Among these, we cite the use of a block-wise embedding scheme in which each block has a different embedding strength value according to some HVS-based features. We are investigating the contrast visibility factor to drive such a scheme

3.2 Watermark Decoding

To extract the hidden bits, we need to determine the slope the Hermite spline by cubic interpolator. To generate the cubic interpolator, the slopes { $\overline{x}_i$ } must be determined. One common way to specify the slopes is to require second derivative (C^2) continuity. The resulting system of tri-diagonal equations can then be solved using various iterative or direct methods [9, 10], with the resulting interpolator being known as the full cubic spline. However, this calculation requires all of the points on the curve to be known, and it is thus not sufficiently local to be used for real-time reconstruction.

3.2.1 Extraction Algorithm

We take the received watermarked signal and key for generating the four samples to get two control points. Now we use these points for finding slope of the line which connects them. Once we have the slope of the line, we can generate the line points with the Bresenhams Line drawing Algorithm .Now we can get the spline curve points stored in the points along the line with LSB Algorithm .With these Hermite Spline curve points and control points we can calculate the slopes of the Hermite spline curve at each control point. Here the slopes are nothing but the watermarked bits which we have sent it with Host image signal.

The slopes can also be adjusted interactively. Methods such as Bezier curves [11] have been designed to do this naturally by specifying additional control points. This is not a satisfactory approach if the curve generation is to proceed automatically from a set of sample points without operator intervention. What is required instead is a method to estimate the individual slopes using only a few of the neighbouring points.

The simplest method for determining the slope locally is to use a parabola through a sample point and its left and right neighbours to determine the slope at the point (modifications can be made for the end points of the curve). The parabolic equation

$$g(s) = \frac{x_i}{2}(s - s_i)^2 + x_i(s - s_i) + x_i$$

With $g(s_{i-1]}) = x_{i-1}$ and $g(s_{i+1}) = x_{i+1}$ Can be solved to yield

$$\frac{x_i = d_{i-1}m_{i,j+1} + d_i m_{i-1,i}}{d_{i-1} + d_i}$$

The value m_{ij} is the divided difference of the curve (Figure 8),

The interpolator resulting from this parabolic fit, known as the Bessel interpolator, is quite simple, but does not smooth as well as methods that use more of the neighbouring points. It is, in effect, a four-point interpolator, since the points x_{i-1}, x_i, x_{i+1} and x_{i-2} are needed to define the two end slopes x_i and x_{i+1} for the segment $f_i(s)$.

Higher-order polynomials (such as quadratics) can be fitted to determine the slope, but the exact solution rapidly becomes complex. An assumption of near-uniform spacing can be used to obtain some simple results which take the form of a SUlTI of divided differences weighted by fixed rational numbers.

4 CONCLUSION

In this paper, we presented a watermarking approach with new algorithms. The algorithm is applied to image signals by using four approximation coefficients of the image blocks which may be selected according to a secret key. We used a simple classical Least Significant Bit (LSB) algorithm for inserting the message bits. In addition to be invariant to the volumetric distortions, several simulations showed that the proposed algorithm is highly robust against common watermarking attacks such as AWGN, compression, and filtering.

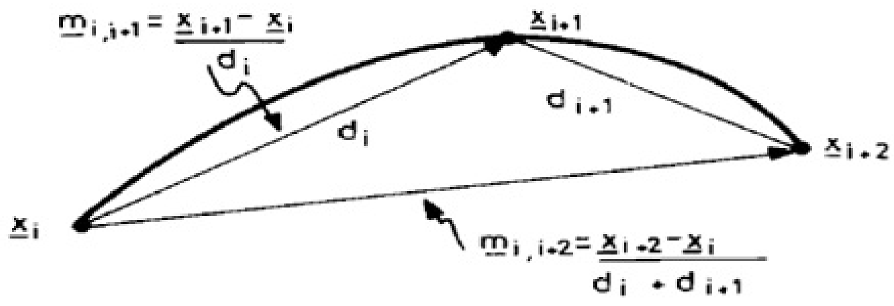

$\underline{m}_{i,i+1} = \frac{\underline{x}_{i+1} - \underline{x}_i}{d_i}$
$\underline{x}_{i+1}$
d_i
d_{i+1}
d_i
$\underline{x}_{i+2}$
$\underline{x}_i$
$\underline{m}_{i,i+2} = \frac{\underline{x}_{i+2} - \underline{x}_i}{d_i + d_{i+1}}$

Figure 8: Divided differences of a curve used to calculate the slopes and are defined as the slope of a line connecting two sample points. [12].

ACKNOWLEDGMENT

We would like to thank all our colleagues who encouraged and helped in completing this work.

REFERENCES

[1] J. Seitzi, *Digital Watermarking For Digital Media*, Arlington, VA, USA, Information Resources Press, 2005.

[2] Allam Mohan, Salina Adinarayana, *A New Blind Image Watermarking using Hermite Spline Approach*, IJCSIT, (2012) 4812-4817.

[3] David G. Lowe, *Distinctive Image Features from Scale-Invariant Keypoints*,International Journal of Computer Vision 60(2), (2004) 91110.

[4] M. A. Akhaee, S. M. E. Sahraeian, B. Sankur, *Robust Scaling-based image watermarking using maximum-likelihood decoder with optimum strength factor*,IEEE Trans. Multimedia, vol. 11, (2009) 822 833.

[5] BOEHM.W, *On cubics: a survey* , ibid, (1982) 201-226.

[6] R. SZELISKI, *Real time coding of hand drawn curves*, University of British Columbia, Vancouver, Canada 1981.

[7] R. S. LAPALME, *An interactive data reduction technique for line drawings*, Royal Military College, Canada,1977.

[8] R. P. DUBE, *Preliminary specification of spline curves*, IEEE Trans, 28 (1979) 286-290

[9] M. L. LIOU, *Spline fit made easy*,IEEE Trans, (1976) 522- 527.

[10] R. MOSS,A. LINGARD, *Parametric spline curves in integer Arithmetic designed for use in microcomputer Controlled plotters*,Computer & Graphics, 4 (1979) 51- 56.

[11] I. Cox, J. Kilian, F. Leighton, and T. Shamoon, *Secure spread spectrum Watermarking for multimedia*,IEEE Trans. Image Process, vol. 6,12 (1997) 1673 1687.

[12] R.Szeliski, *New Hermite cubic interpolator for two-dimensional curve generation*,IEEE Trans.,6 (1986) 341-347.