http://infor.seaninstitute.org/index.php/infokum/index
JURNAL INFOKUM, Volume 10, No.1, Desember 2021 ISSN : 2302-9706

Implementation Of Watermarking Using The Singular Value Decomposition Method

Satria Yudha Prayogi[1], Sony Bahagia Sinaga[2]
[1]Universitas Islam Sumatera Utara, Jl. Sisingamarangaraja, Medan, 20217, Indonesia
[2]AMIK STIEKOM Sumatera Utara, Jl. Adam Malik No. 28, Rantauprapat, Indonesia
Email : satria.yp@ft.uisu.ac.id, sonybahagia@gmail.com

Article Info
Received 30 November 2021
Revised 19 December 2021
Accepted 29 December 2021

Technological developments have made it easy for the distribution of digital data (such as text, images, audio, and video) both with disk media and through internet media. On the other hand, this convenience can cause problems with copyright protection. One technique to overcome this problem is to insert a sign (watermark) as proof of ownership into digital data known as watermarking. Watermarks can be inserted into the spatial domain or the frequency domain. In general, watermarking in the frequency domain is more resistant to attacks than watermarking in the spatial domain. The watermarking method used in this study uses a transformation domain, namely Singular Value Decomposition (SVD). Tests used to provide a blurring effect). Where the technique using Singular Value Decomposition (SVD) is generally inserted into singular values based on the consideration that the singular value will not experience significant changes if there is a slight disturbance in the image.

Keywords: Watermark, SVD (Singular Value Decompotition)

1. Introduction

Steganography is the art of hiding a secret message (hiding message) in digital media in such a way that someone does not realize there is a message in the media (ANDI, Digital Image Processing, 2009). Watermarking is a technique used to insert a small amount of information that indicates ownership or other data in multimedia material, but its existence is not known to the human senses and is able to withstand various attacks that intend to remove the inserted information. To answer the need for copyright protection, watermarking technology is used to protect data and information copyright. Watermarking is a steganography application. However, watermarking itself has several differences from steganography. Steganography aims to send any secret message without raising suspicion where the container media has no meaning (meaningless). The requirements of steganography are safe, difficult to detect, and contain as many messages as possible (large capacity). While watermarking aims for copyright protection (copyright), proof of ownership (ownership), and fingerprints (fingerprinting) where it is the storage media that is given protection. The requirements that watermarking must have are not only difficult to detect, but also must be resistant and difficult to remove (robustness). One method of watermarking is to use the SVD (Singular Value Decomposition) method. Where the technique using Singular Value Decomposition (SVD) is generally inserted into singular values based on the consideration that the singular value will not experience significant changes if there is a slight disturbance in the image.

2. Method

2.1 Table

There are two types of 8-bit color, namely 8-bit color images using a color palette of 256 with each palette having a specific RGB colormap. This model is used more often. Second each pixel has an 8 bit format as follows.

Table 1. Bit Truecolor

Bit-7	*Bit-6*	*Bit-5*	*Bit-4*	*Bit-3*	*Bit-2*	*Bit-1*	*Bit-0*
R	R	R	G	G	G	B	B

A 16 bit color image is usually referred to as a (highcolor image) with each pixel represented by 2 bytes of memory (16 bits). 16-bit color has 65,536 colors, in its bit formation, the red and blue values take place in the right and left 5 bits. The green component has 5 bits plus 1 extra bit.

Table 2. 16 Bit Color Series

Bit-15	*Bit-14*	*Bit-13*	*Bit-12*	*Bit-11*	*Bit-10*	*Bit-9*	*Bit-8*	*Bit-7*	*Bit-6*	*Bit-5*	*Bit-4*	*Bit-3*	*Bit-2*	*Bit-1*	*Bit-0*

2.2 Graphic Content

A digital image can be defined as a function of two variables, f(x,y), where x and y are spatial coordinates and the value f(x,y) is the intensity of the image at those coordinates, this is illustrated in the figure below. The basic technology for creating and displaying color in digital images is based on research that a color is a combination of three basic colors, namely red, green, and blue (Red, Green, Blue-RGB).

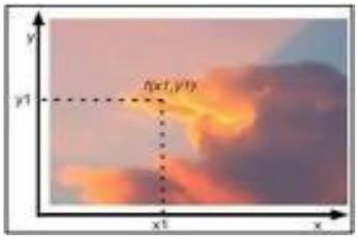

Figure 1 Digital Image

Color images, which are usually RGB images, are stored in a matrix of size m x n, each of which defines red, green and blue colors for each pixel.

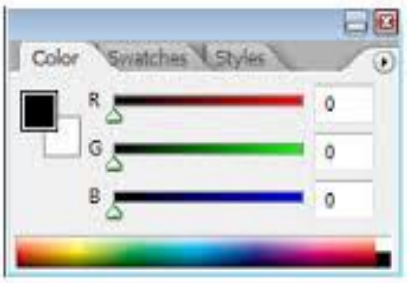

Figure 2 RGB Components and Composition

The color of light that is responded to by the eye is visible light (visible spectrum) with wavelengths ranging from 400 nanometers (blue) to 700 nanometers (red). For example, an object that reflects blue light (wavelength 450 - 490) and absorbs other light will turn blue.

Figure 3 Color Image Storage In Computer Memory

3 Results and Discussion

3.1 Mathematical Equations

In general, the singular value decomposition algorithm, in the insertion process is as follows:

1. Called matrix A^TA with value eigen λ_i for each 1≤ i ≤ then the value of the singular matrix A that is $\sigma_i = \sqrt[2]{\lambda_I}$ Equality (1)
2. Form a diagonal matrix $S = \begin{matrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_n \end{matrix}$ Equality (2)
3. Find the set of eigenvectors from the matrix A^TA . Suppose $\{v_i, v_2, v_3, \ldots v_n\}$ are eigenvectors corresponding to the value of λ_i Equality (3)
4. An orthogonal matrix is formed V = $[v_1, v_2, v_3, \ldots..v_n]$ Equality (4)
5. Form a vector set $\{u_1, u_2, \ldots u_n\}$ with $u_i = \frac{1}{\sigma_i} Av_i$ for each 1≤ i ≤n Equality (5)
6. Form an orthogonal matrix U = $[u_1, u_2, u_n]$ Equality (6)
7. Bentuk dekomposisi SVD adalah $A = US_wV^T$ Equality (7)
   The singular value S is then added to the product of the watermark W with an alpha value
8. S_t = S + nilai alfa * W Equality (8)
   Where the alpha value is the intensity that determines the strength of the watermark to be inserted. Then do the decomposition on S_t.
9. $S_t = U_wS_wV_w^T$ Equality (9)
   As a final step, the obtained Sw is then used to form a watermarked image along with the U and V matrices of the original image.

10. $A_w = US_wV^T$ Equality (10)
11. The steps in the extraction process are as follow $W = \frac{S_t - S}{nilai\ alfa}$ Equality (11)

4. Programming Code

```
Class file {
u4 magic;
u2 minor_version;
u2 major_version;
u2 constant_pool_count;
cp_info constant_pool [constant_pool_count-1];
u2 access_flags;
u2 this_class;
u2 super_class;
u2 interfaces_count;
u2 interfaces [interfaces_count];
u2 fields_count;
field_info fields[fields_count];
u2 methods_count;
method_info methods [methods_count];
u2 attributes_count; attribute_info attributes [attributes_count];
}
cp_info {
u1 tag; u1 info[ ];
}
Tag
CONSTANT_Class              7
CONSTANT_Fieldref           9
CONSTANT_Methodref              10
CONSTANT_InterfaceMethodref     11
CONSTANT_String             8
CONSTANT_Integer            3
CONSTANT_Float              4
CONSTANT_Long                   5
CONSTANT_Double             6
CONSTANT_NameAndType        12
CONSTANT_Utf8                   1
CONSTANT_Class_info {
u1 tag;
u2 name_index;
}
CONSTANT_Fieldref_info {
u1 tag;
u2 class_index;
u2 name_and_type_index;
}
```

```
CONSTANT_Methodref_info {
u1 tag;
u2 class_index;
u2 name_and_type_index;
}
CONSTANT_InterfaceMethodref_info {
u1 tag;
u2 class_index;
u2 name_and_type_index;
}
CONSTANT_String_info {
u1 tag;
u2 string_index;
}
CONSTANT_Integer_info {
u1 tag;
u4 bytes ;
}
CONSTANT_Float_info {
u1 tag;
u4 bytes;
}
CONSTANT_Long_info {
u1 tag;
u4 high_bytes;
u4 low_bytes;
}
CONSTANT_Double_info {
u1 tag;
u4 high_bytes;
u4 low_bytes;
}
CONSTANT_NameAndType_info {
u1 tag;
u2 name_index;
u2 descriptor_index;
}
CONSTANT_Utf8_info {
u1 tag;
u2 length;
u1 bytes[length];
}
access_flag
ACC_PUBLIC 0x0001
ACC_FINAL 0x0010
ACC_SUPER 0x0020
ACC_INTERFACE 0x0200
```

ACC_ABSTRACT 0x0400
field_info {
u2 access_flags;
u2 name_index;
u2 descriptor_index;
u2 attributes_count; attribute_info attributes [attributes_count];
}
access_flag
ACC_PUBLIC 0x0001
ACC_PRIVATE 0x0002
ACC_PROTECTED 0x0004
ACC_STATIC 0x0008
ACC_FINAL 0x0010
ACC_VOLATILE 0x0040
ACC_TRANSIENT 0x0080
method_info {
u2 access_flags;
u2 name_index;
u2 descriptor_index;
u2 attributes_count; attribute_info attributes [attributes_count];
}
access_flag
ACC_PUBLIC 0x0001
ACC_PRIVATE 0x0002
ACC_PROTECTED 0x0004
ACC_STATIC 0x0008
ACC_FINAL 0x0010
ACC_SYNCHRONIZED 0x0020
ACC_NATIVE 0x0100
ACC_ABSTRACT 0x0400
ACC_STRICT 0x0800
attribute_info {
u2 attribute_name_index;
u4 attribute_length;
u1 info[attribute_length];
}
ConstantValue_attribute {
u2 attribute_name_index;
u4 attribute_length;
u2 constantvalue_index;
}
Code_attribute {
u2 attribute_name_index;
u4 attribute_length;
u2 max_stack;
u2 max_locals;
u4 code_length;

u1 code[code_length];
u2 exception_table_length; {
u2 start_pc; u2 end_pc;
u2 handler_pc;
u2 catch_type;
} exception_table [exception_table_length];
u2 attributes_count;
attribute_info attributes [attributes_count];
}
Synthetic_attribute {
u2 attribute_name_index;
u4 attribute_length;
}
Exceptions_attribute {
u2 attribute_name_index;
u4 attribute_length;
u2 number_of_exceptions;
u2 exception_index_table [number_of_exceptions];
}
InnerClasses_attribute {
u2 attribute_name_index;
u4 attribute_length;
u2 number_of_classes; {
u2 inner_class_info_index;
u2 outer_class_info_index;
u2 inner_name_index;
u2 inner_class_access_flags;
} classes [number_of_classes];
}
inner_class_access_flags
sACC_PUBLIC 0x0001
ACC_PRIVATE 0x0002
ACC_PROTECTED 0x0004
ACC_STATIC 0x0008
ACC_FINAL 0x0010
ACC_INTERFACE 0x0200
ACC_ABSTRACT 0x0400
SourceFile_attribute {
u2 attribute_name_index;
u4 attribute_length;
u2 sourcefile_index;
}
LineNumberTable_attribute {
u2 attribute_name_index;
u4 attribute_length;
u2 line_number_table_length;
{ u2 start_pc; u2 line_number;

```
} line_number_table [line_number_table_length];
}
LocalVariableTable_attribute {
u2 attribute_name_index;
u4 attribute_length;
u2 localvariable_table_length;
{ u2 start_pc;
u2 length;
u2 name_index;
u2 descriptor_index;
u2 index;
} local_variable_table [localvariable_table_length];
}
Deprecated_attribute {
u2 attribute_name_index;
u4 attribute_length;
}
```

Reference

[1]. Jogiyanto, 2005. *Analisis dan Desain*. Yogyakarta : Andi
[2]. T.Sutoyo, Edy Mulyanto, Vincent Suhartono, Oky Dwi Nurhayati, Wijanarto. *Teori Pengolahan Citra Digital*: 2009.
[3]. Abdul Kadir. *Dasar Pengolahan Citra Dengan Delphi*: 2009
[4]. Prahadi Dagdoyo. *Aplikasi Watermark Pada Citra Digital Menggunakan Metode Singular Value Decomposition*: 2011
[5]. Abdul Kadir & Adhi Susanto. *Pengolahan citra digital*: 2013
[6]. Rachmad Hakim S. 2002. *Visual Basic 2008*. Jakarta: PT Elex Media Komputindo.
[7]. Adi Nugroho.2006. *Analisis dan Perancangan Sistem Informasi dengan Metodologi Berorientasi Objek*.
[8]. Rodia. 2004. *Watermarking Sebagai Teknik Penyembunyian Label Hak Cipta Pada Data Digital Menggunakan Algoritma DCT (Discrete Cosine Transform).* Jakarta: Universitas Gunadarma.
[9]. Alinurdin, Lili. 2006. *Gambar Raster (Bitmap Image)*. CIC Group. Barni, M., Bartolini, Fahmi. 2007. *Studi dan Implementasi Watermarking Citra Digital Dengan Menggunakan Fungsi Hash*. Bandung : Institut Teknologi Bandung.