DAFTAR PUSTAKA

Agung, Wiseto. 2007. Metode *Assymetric Watermarking* pada Citra Digital Berbasiskan pada Permutasi-RC4 dan Fungsi Chaos, (www.*informatika.stei.itb.ac.id* , diakses 11 Februari 2010)

Armyta Dini. 2006. Makalah Studi Mengenai Aplikasi Steganography Camouflage Beserta Pemecahan Algoritmanya (www. *informatika.stei.itb.ac.id , diakses 3 Maret 2010)*

Baldman. 2003. Bit error ratio testing: How many bits are enough? *(*https://www.iol.unh.edu *, diakses July 2012)*

Barata, A.I. 2005. Watermarking dengan Algoritma Kunci Publik untuk Verifikasi dan Otentikasi Citra**,** (*http://www.informatika.org* Diakses pada bulan 27 Desember 2009).

Cahyana et al. 2007. Teknik Watermarking Citra Berbasis SVD. Proceeding of National Conference Computer Science and Information Technology 2007. Hal 294-298

Cox, I., J., Kilian, J., Leighton, F., T., dan Shamoon, T. 1997. Secure Spread Spectrum Watermarking for Multimedia, *Proc. IEEE Transactions on Image Processing*, Vol. 6, No. 12, pp 1673-1687. (*ftp://ftp.nj.nec.com/pub/ingemar/papers/ip97.zip* , diakses 25 April 2012)

Fajri, 2008, Bab 3 Desain dan Implementasi. (http://fajri.freebsd.or.id. , diakses 10 Januari 2010)

FIPS2002 Federal Information Processing Standards Publications 180-2 2002 **Secure Hash Standard**, USA: National Institute of Standards and Technology.

J. Fridrich 2002. Lossless Data Embedding - New Paradigm in Digital Watermarking : EURASIP Journal on Applied Signal Processing 2002:2, 185–196

Johnson, N., F., dan Jajodia, S. 1998. Exploring Staganography: Seeing the Unseen, *IEEE Computer*, pp 26-34. (*http://isse.gmu.edu/~njohnson/pub/r2026.pdf* , diakses 13 February 2010)

Mahyuddin, Rezza . *2009.* Studi dan Perbandingan Metodologi Autentikasi Gambar Berbasis Digital Signature: Jurnal *MakalahIF3058-2009-b007.pdf.* (*www. informatika.stei.itb.ac.id, diakses 13 February 2010)*

Munir, Rinaldi. 2004. Pengolahan Citra Digital Dengan Pendekatan Algoritmik. Informatika, Bandung

Munir, Rinaldi. Diktat Kuliah IF5054 Kriptografi. 2006. Bandung: Institut Teknologi Bandung. (www.itb.ac.id, diakses Maret 2010)

Ni et al . 2003. Reversible Data Hiding," in *International Symposium on Circuits and Systems, Proc. IEEE* 2, 912-915

Persada. 2009. Studi Dan Implmentasi Non Blind Watermarking Dengan Metode Spread Spectrum (www.informatika.stei.ac.id, diakses 2 Maret 2010)

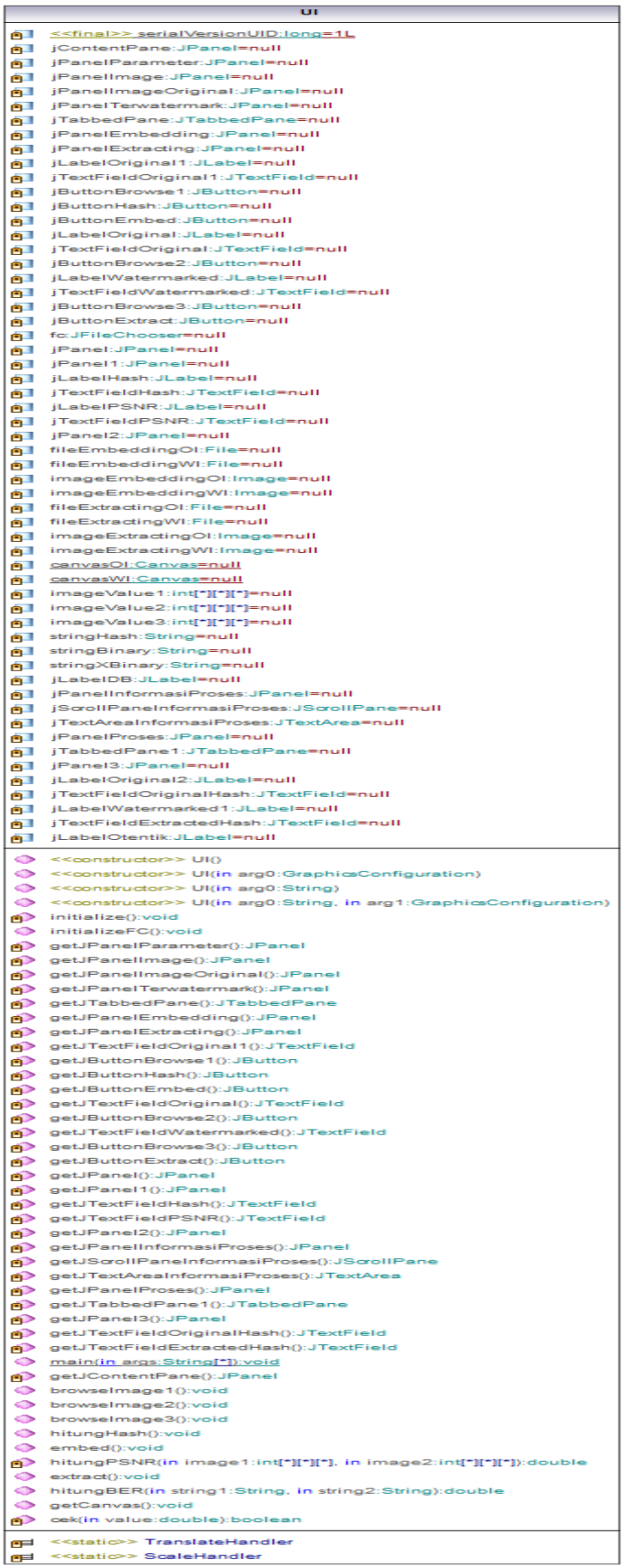
Rachmawati. 2008*.* Estimasi parameter geometris benda berbasis pengolahan citra digital *(estimation of geometric object parameters based on Digital image processing).* **(**http://digilib.ittelkom.ac.id , diakses July 2012)

Sebastian. A ,2007, Implementasi dan Perbandingan Performa Algoritma Hash SHA-1, SHA-256, dan SHA-512 (http://www.informatika.org. , diakses 25 Januari 2010)

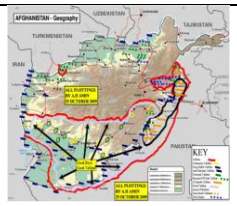
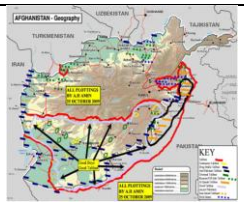
Sutoyo, Edy Mulyanto, Suhartono.V, 2009. Teori Pengolahan Citra Digital, Penerbit Andi Yogyakarta.

LAMPIRAN 1

Gambar Kelas UI



LAMPIRAN 2

Citra Asli	Citra Terwatermark
1.  Airplane.jpg (169,1 KB)	 Watermarked_Airplane.jpg (20,3 KB)
2.  Foot.jpg (154,8 KB)	 Watermarked_Foot (22,4 KB)
3.  Tree.jpg (308,6 KB)	 Watermarked_Tree (71,2 KB)
4.  Taliban_locs.jpg (407,7 KB)	 Watermarked_Taliban_locs.jpg (79,6 KB)
5.  Lena.jpg (66,1 KB)	 Watermarked_Lena.jpg (43,4 KB)
6.  Butterfly.jpg (302,5 KB)	 Watermarked_Butterfly.jpg (38,9 KB)

LAMPIRAN 3

Coding Untuk Read Image

```
package com.dct;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;

import javax.imageio.ImageIO;

public class ImageReader {

	private File file		=null;
	private BufferedImage image	=null;
	private int[][][]imageValue	=null;

	public ImageReader(File file) {
		super();
		this.file = file;
		initialize();
	}//end of constructor

	private void initialize() {
		if(this.file!=null){
			try {
				this.image=ImageIO.read(this.file);
				int width	=image.getWidth();
				int height	=image.getHeight();
				this.imageValue	=new int[width][height][3];
				for(int i=0 ; i<width ; i++){
					for(int j=0 ; j<height ; j++){
						int color	= image.getRGB(i,j);
						int red		= (color & 0x00ff0000) >> 16;
						int green	= (color & 0x0000ff00) >> 8;
						int blue	= (color & 0x000000ff);

						this.imageValue[i][j][0]=red;

						this.imageValue[i][j][1]=green;

						this.imageValue[i][j][2]=blue;
					}//end of for(int j=0 ; j<height ; j++)
				}//end of for(int i=0 ; i<width ; i++)
			} catch (IOException e) {
				e.printStackTrace();
			}//end of try catch
			System.out.println("reading the data");
		}//end of if(this.file!=null)
	}//end of initialize()

	public BufferedImage getImage() {
		return image;
	}

	public int[][][] getImageValue() {
		return imageValue;
```

```
	}

}//end of ImageReader
```

Coding Untuk Write Image

```
package com.dct;
import java.awt.Image;
import java.awt.image.BufferedImage;
import java.awt.image.WritableRaster;
import java.io.File;
import java.io.IOException;

import javax.imageio.ImageIO;
import javax.swing.ImageIcon;


public class ImageWriter {
	private int[][][]imageValue	= null;
	private File file		= null;
	private Image image		= null;

	public ImageWriter(int[][][] imageValue, File file) {
		super();
		this.imageValue = imageValue;
		this.file = file;
		this.initialize();
	}//end of constructor

	public Image getImage() {
		return image;
	}

	public int getImageSize(){
		if(this.imageValue!=null){
			return
(this.imageValue.length*this.imageValue[0].length*this.imageValue[0][0].length);
		}else{
			return 0;
		}
	}

	private void initialize() {
		if(this.file!=null&&this.imageValue!=null){
			try {
				int width    =imageValue.length;
				int height   =imageValue[0].length;
				BufferedImage image = new
BufferedImage(width,height,BufferedImage.TYPE_INT_RGB);
				WritableRaster raster = image.getRaster();
				for(int i=0;i<width;i++){
					for(int j=0;j<height;j++){
						int[]
px={imageValue[i][j][0],imageValue[i][j][1],imageValue[i][j][2]};
						raster.setPixel(i,j,px);
					}//end of for(int j=0;j<height;j++)
				}//end of for(int i=0;i<width;i++)
				ImageIO.write(image,"JPG",this.file);
```

```
                        this.image=new
ImageIcon(this.file.getPath()).getImage();
                    } catch (Exception e) {
                        e.printStackTrace();
                    }//end of try catch
                }//end of if
        }//end if initialize()



}//end of ImageWriter
```

Hitung Hash

```
package com.dct;
import java.awt.image.BufferedImage;
import java.io.ByteArrayOutputStream;
import java.io.File;
import java.security.MessageDigest;

import javax.imageio.ImageIO;

public class ImageHash {

        public ImageHash(File fileimage){
                super();
                updateImageHash(fileimage);
        }//end of constructor

        private static String stringHash;
        private static String stringBinary;
        private static String stringXBinary;

        public static String getStringHash() {
                return stringHash;
        }

        public static String getStringBinary() {
                return stringBinary;
        }

        public static String getStringXBinary() {
                return stringXBinary;
        }

        public void updateImageHash(File input){
                try {
                        BufferedImage buffImg = ImageIO.read(input);
                        ByteArrayOutputStream outputStream = new
ByteArrayOutputStream();
                  ImageIO.write(buffImg, "jpg", outputStream);
                  byte[] data = outputStream.toByteArray();
                        MessageDigest md = MessageDigest.getInstance("SHA-
384");//calculate Hash
                        md.update(data);
                        byte[] hash = md.digest();
                  updateStringHash(hash);
                } catch (Exception e) {
```

```
                    // TODO Auto-generated catch block
                    e.printStackTrace();
             }
       }//end of

       static void updateStringHash(byte[] inBytes) {
              String hexString ="";
        String binString ="";
        String xbinString ="";
        for (int i=0; i < inBytes.length; i++) {
            hexString +=Integer.toString( ( inBytes[i] & 0xff ) + 0x100,
16).substring( 1 );
            binString +=Integer.toBinaryString(( inBytes[i] & 0xff ) +
0x100).substring( 1 );
         }
       stringHash          =hexString;
       stringBinary =binString;
       xbinString          =binString.replaceAll("0", "-1");
       stringXBinary=xbinString;
       }//end of updateStringHash(byte[] inBytes)

}//end of class ImageHash
```

Coding Untuk Perhitungan DCT

```
       //-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -
- -- -- -- -- -- -- -- -- --
       public double[][]getDCT(double[][]value, int m, int n){
              double[][] dct=null;
              if((value!=null)&&(m>0)&&(n>0)){
                     int width=value.length;
                     int height=value[0].length;
                     int nBlockM=(int) Math.floor((double)width/(double)m);
                     int nBlockN=(int) Math.floor((double)height/(double)n);
                     dct=new double[nBlockM*m][nBlockN*n];
                     for(int i=0;i<nBlockM;i++){
                            for(int j=0;j<nBlockN;j++){
                                   int po=m*i;
                                   int qo=n*j;
                                   double[][] tempValue=new double[m][n];
                                   //inisialisasi nilai tempRGB
                                   for(int p=0;p<m;p++){
                                          for(int q=0;q<n;q++){
                                                 int baris=po+p;
                                                 int kolom=qo+q;
                                                 tempValue[p][q]=value[baris][kolom];
                                          }//end of for q
                                   }//end of for p
                                   //hitung dctRGB dari tempRGB
                                   double[][] dctValue=computeDCT(tempValue);
                                   //meletakkan nilai dctR, dctG dan dctB ke dctRGB
                                   for(int p=0;p<m;p++){
                                          for(int q=0;q<n;q++){
                                                 int baris=po+p;
                                                 int kolom=qo+q;
                                                 dct[baris][kolom]=dctValue[p][q];
                                          }//end of for q
                                   }//end of for p
```

```
                        }//end of for j
                }//end of for i
        }//end of if((value!=null)&&(m>0)&&(n>0))
        return dct;
    }//end of getDCT()

    public double[][]computeDCT(double[][] value){
        double[][] dct=null;
        if(value!=null){
            int m,n,p,q;
            m=value.length;
            p=m;
            n=value[0].length;
            q=n;
            dct=new double[p][q];
            double alpha_p,alpha_q;
            for(int i=0;i<dct.length;i++){

    if(i==0){alpha_p=(double)((double)1/Math.sqrt((double)m));}
                    else{alpha_p=Math.sqrt((double)((double)2/(double)m));}
                    for(int j=0;j<dct[i].length;j++){

    if(j==0){alpha_q=(double)((double)1/Math.sqrt((double)n));}

    else{alpha_q=Math.sqrt((double)((double)2/(double)n));}
                            //--- --- --- --- --- --- ---
                            double sigma=0;
                            for(int k=0;k<m;k++){
                                for(int l=0;l<n;l++){
                                    double
arg1=(Math.PI*(double)(2*k+1)*(double)i)/(double)(2*m);
                                    double
arg2=(Math.PI*(double)(2*l+1)*(double)j)/(double)(2*n);

    sigma=sigma+value[k][l]*Math.cos(arg1)*Math.cos(arg2);
                                    }//end of for for(int l=0;l<n;l++)
                            }//end of for(int k=0;k<m;k++)
                            double sigma1=alpha_p*alpha_q*sigma;
                            dct[i][j]=sigma1;
                            //--- --- --- --- --- --- ---
                    }//end of for(int j=0;j<dct[i].length;j++)
            }//end of for(int i=0;i<dct.length;i++)
        }//end of if(imageValue!=null)
        return dct;
    }//end of computeDCT()

    //-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -
- -- -- -- -- -- -- -- -- --
```

Coding Untuk Perhitungan IDCT

```
//-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
-- -- -- -- -- -- --
    public double[][]getIDCT(double[][]dct, int m, int n){
        double[][] idct=null;
        if((dct!=null)&&(m>0)&&(n>0)){
            int width=dct.length;
            int height=dct[0].length;
            idct=new double[width][height];
```

```
                int nBlockM=width/m;
                int nBlockN=height/n;
                for(int i=0;i<nBlockM;i++){
                        for(int j=0;j<nBlockN;j++){
                                int po=m*i;
                                //int pi=m+po;
                                int qo=n*j;
                                //int qi=n+qo;
                                double[][] dctValue=new double[m][n];
                                //inisialisasi nilai dctRGB
                                for(int p=0;p<m;p++){
                                        for(int q=0;q<n;q++){
                                                int baris=po+p;
                                                int kolom=qo+q;
                                                dctValue[p][q]=dct[baris][kolom];
                                        }//end of for q
                                }//end of for p
                                //hitung idctRGB dari dctRGB
                                double[][] idctValue=computeIDCT(dctValue);
                                //meletakkan idctR, idctG dan idctB ke idctRGB
                                for(int p=0;p<m;p++){
                                        for(int q=0;q<n;q++){
                                                int baris=po+p;
                                                int kolom=qo+q;
                                                idct[baris][kolom]=idctValue[p][q];
                                        }//end of for q
                                }//end of for p
                        }//end of for j
                }//end of for i
        }//end of if((dct!=null)&&(m>0)&&(n>0))
        return idct;
}//end of getIDCT()

public double[][]computeIDCT(double[][] dctValue){
        double[][]idctValue=null;
        if(dctValue!=null){
                int m,n,p,q;
                p=dctValue.length;
                m=p;
                q=dctValue[0].length;
                n=q;
                idctValue=new double[m][n];
                double alpha_p,alpha_q;
                for(int i=0;i<idctValue.length;i++){
                        for(int j=0;j<idctValue[i].length;j++){
                                double sigma=0;
                                for(int k=0;k<p;k++){

if(k==0){alpha_p=(double)((double)1/Math.sqrt((double)m));}

else{alpha_p=Math.sqrt((double)((double)2/(double)m));}
                                        for(int l=0;l<q;l++){

if(l==0){alpha_q=(double)((double)1/Math.sqrt((double)n));}

else{alpha_q=Math.sqrt((double)((double)2/(double)n));}
                                                //--- --- --- --- --- --- ---
                                                double
arg1=(Math.PI*(double)(2*i+1)*(double)k)/(double)(2*m);
```

```
                                                double
arg2=(Math.PI*(double)(2*j+1)*(double)l)/(double)(2*n);

      sigma=sigma+alpha_p*alpha_q*dctValue[k][l]*Math.cos(arg1)*Math.cos(arg2);
                                                //--- --- --- --- --- --- ---
                                          }//end of for(int l=0;l<q;l++)
                                    }//end of for(int k=0;k<p;k++)
                                    idctValue[i][j]=sigma;
                              }//end of for(int j=0;j<imageValue[i].length;j++)
                        }//end of for(int i=0;i<imageValue.length;i++)
                  }//end of if(dctValue!=null)
                  return idctValue;
      }//end of computeIDCT(double[][] dctValue)

      //-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -
- -- -- -- -- -- -- -- -- --
```

Coding Untuk Penyisipan Nilai Hash

```
//-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
-- -- -- -- -- -- --
      public double[][]embed(double[][]value, String watermark, double alpha, int
m, int n){
            double[][]watermarkedValue=null;
            if((value!=null)&&(watermark!=null)&&(m>0)&&(n>0)){
                  watermarkedValue=new double[value.length][value[0].length];
                  watermarkedValue=copy(value);
                  double[][] dct=getDCT(value, m, n);
                  int width=dct.length;
                  int heigh=dct[0].length;
                  int blockM=width/m;
                  int blockN=heigh/n;
                  int totalBlock=blockM*blockN;
                  int sebaranWatermark=(int)
Math.ceil((double)watermark.length()/(double)totalBlock);

                  int[]intWatermark=new int[sebaranWatermark*totalBlock];
                  for(int i=0;i<intWatermark.length;i++){
                        int index0=i,index1=1+index0;
                        String s="17";//17 hanya angk pengeculian selain 0 dan
1
                        if(index1<=watermark.length()){
                              s=watermark.substring(index0,index1);
                              //System.out.print(s);
                        }//end if(index1<=watermark.length())
                        int is=Integer.parseInt(s.trim());
                        if(is==1){intWatermark[i]=1;}
                        else if(is==0){intWatermark[i]=-1;}
                        else{intWatermark[i]=0;}
                  }//end of for i

                  double[][]tempWatermarked=new double[width][heigh];
                  tempWatermarked=copy(dct);
                  int in=0;
                  Random acak=new Random();
                  for(int i=0;i<blockM;i++){
                        for(int j=0;j<blockN;j++){
                              int mo=m*i;
                              int no=n*j;
```

```
                                    //inisialisasi block
                                    double[][]block=new double[m][n];
                                    for(int k=0;k<m;k++){
                                          for(int l=0;l<n;l++){
                                                int baris=mo+k;
                                                int kolom=no+l;
                                                block[k][l]=dct[baris][kolom];
                                          }//end of for l
                                    }//end of for k

                                    //cari nilai DCT tengah sebanyak sebaranWatermark
                                    int[][]status=getPositionOfMiddleValue(block,
sebaranWatermark);

                                    //sisipkan watermarking
                                    int insert=0;
                                    for(int k=0;k<m;k++){
                                          for(int l=0;l<n;l++){

      if((status[k][l]==1)&&(insert<sebaranWatermark)){

      /////System.out.print(intWatermark[in]);
                                                      double
w=(acak.nextDouble())*(double)(intWatermark[in]);

                                                      double c0=block[k][l];
                                                      double
c1=c0*((double)1+alpha*w);
                                                      int baris=mo+k;
                                                      int kolom=no+l;
                                                      /////System.out.print(" -->
Before: "+(tempWatermarked[baris][kolom]));

      tempWatermarked[baris][kolom]=c1;

                                                      /////System.out.print("
After: "+(tempWatermarked[baris][kolom])+"\n");
                                                      in++;
                                                      insert++;

                                                }//end of
if((status[k][l]==1)&&(insert<sebaranWatermark))
                                                if(insert==sebaranWatermark){break;}
                                          }//end of for l
                                          if(insert==sebaranWatermark){break;}
                                    }//end of for k
                                    if(intWatermark[in]==0){break;}
                              }//end of for(int j=0;j<nblockN;j++)
                              if(intWatermark[in]==0){break;}
                        }//end of for(int i=0;i<nBlockM;i++)

                        double[][] idct=getIDCT(tempWatermarked, m, n);

                        for(int i=0;i<idct.length;i++){
                              for(int j=0;j<idct[i].length;j++){
                                    watermarkedValue[i][j]=idct[i][j];
                              }//end of for j
                        }//end of for i
```

```
		}//end of if((value!=null)&&(watermark!=null)&&(m>0)&&(n>0))
		return watermarkedValue;
	}//end of embed(double[][]value, String watermark, int m, int n)

	private int[][]getPositionOfMiddleValue(double[][]block, int sebaran){
		int[][]position=null;
		if((block!=null)&&(sebaran>0)){
			position=new int[block.length][block[0].length];
			double[] newValue=reshape(block);
			for(int i=0;i<newValue.length;i++){
				newValue[i]=Math.abs(newValue[i]);
			}//end of for(int i=0;i<newValue.length;i++)
			double[]sortedValue=sortAsc(newValue);
			int begin=1;
			if(sortedValue.length%2==1){
				int mid=(int)
Math.ceil((double)sortedValue.length/(double)2);
				if(sebaran%2==1){
					begin=mid-(int)
(Math.floor((double)sebaran/(double)2));
				}else{
					begin=mid-((sebaran/2)-1);
				}//end of if(sebaran%2==1) else
			}else{
				int mid=sortedValue.length/2;
				if(sebaran%2==1){
					begin=mid-(int)
(Math.floor((double)sebaran/(double)2));
				}else{
					begin=mid-((sebaran/2)-1);
				}//end of if(sebaran%2==1) else
			}//end of if(sortedValue.length%2==1) else
			int lo=begin;
			int up=sebaran+begin;
			for(int h=lo;h<up;h++){
				boolean found=false;
				double key=sortedValue[h];
				for(int i=0;i<position.length;i++){
					for(int j=0;j<position[i].length;j++){

	if((i!=0)&&(j!=0)&&(position[i][j]!=1)&&(Math.abs(block[i][j])==key)){
							position[i][j]=1;
							found=true;
							break;
						}//end of if
					}//end of for j
					if(found==true){break;}
				}//end of for i
			}//end of for(int h=lo;h<up;h++)
		}//end of if((block!=null)&&(sebaran>0))
		return position;
	}//end of getPositionOfMiddleValue(double[][]block, int sebaran)

	private double[]reshape(double[][] value){
		double[] newValue=new double[(value.length*value[0].length)];
		int n=0;
		for(int i=0;i<value.length;i++){
			for(int j=0;j<value[0].length;j++){
				newValue[n]=value[i][j];
```

```
                        n++;
                    }//end of for(int j=0;j<value[0].length;j++)
                }//end of for(int i=0;i<value.length;i++)
                return newValue;
        }//end of reshape(double[][] value)

        private double[]sortAsc(double[] value){
                double[] newValue=copy(value);
                for(int i=1;i<(-1+newValue.length);i++){
                        for(int j=(1+i);j<newValue.length;j++){
                                if(newValue[i]>newValue[j]){
                                        //tukarkan nilainya
                                        double temp=newValue[i];
                                        newValue[i]=newValue[j];
                                        newValue[j]=temp;
                                }//end of if(newValue[i]>newValue[j])
                        }//end of for(int j=(1+i);j<newValue.length;j++)
                }//end of for(int i=0;i<(-1+newValue.length);i++)
                return newValue;
        }//end of sort(double[] value)
```

Coding Untuk Pengekstrakan Nilai Hash

```
public String extract(double[][]valueOriginal, double[][]valueWatermarked, int
watermarkLength, int m, int n){
                StringBuffer tempWatermark=new StringBuffer();
                String watermark="";
                if((valueOriginal!=null)&&(valueWatermarked!=null)&&(m>0)&&(n>0)){
                        double[][] dct1=getDCT(valueOriginal, m, n);
                        double[][] dct2=getDCT(valueWatermarked, m, n);

                        int width1=dct1.length;
                        int heigh1=dct1[0].length;

                        int width2=dct2.length;
                        int heigh2=dct2[0].length;

                        ////////if((width1==width2)&&(heigh1==heigh2)){
                                int width=width1;
                                int heigh=heigh1;

                                int blockM=width/m;
                                int blockN=heigh/n;

                                int out=0;
                                boolean finish=false;
                                for(int i=0;i<blockM;i++){
                                        for(int j=0;j<blockN;j++){
                                                int mo=m*i;
                                                int no=n*j;

                                                //inisialisasi block
                                                //double[][]block1=new double[m][n];
                                                //double[][]block2=new double[m][n];
                                                /////////System.out.println("Next Block
:");
                                                for(int k=0;k<m;k++){
                                                        for(int l=0;l<n;l++){
```

```
                                                int baris=mo+k;
                                                int kolom=no+l;

        //block1[k][l]=dct1[baris][kolom];

        //block2[k][l]=dct2[baris][kolom];

        /////////System.out.println((dct1[baris][kolom])+" <-> 
"+(dct2[baris][kolom]));

        if(dct1[baris][kolom]!=dct2[baris][kolom]){

        if(dct1[baris][kolom]>dct2[baris][kolom]){

        tempWatermark.append("0");

        ////System.out.print("0");
                                                        }else{

        tempWatermark.append("1");

        ////System.out.print("1");
                                                        }//end of 
if(dct1[baris][kolom]!>dct2[baris][kolom])else
                                                        out++;
                                                }//end of 
if(dct1[baris][kolom]!=dct2[baris][kolom])
                                                if(out==watermarkLength){
                                                        finish=true;
                                                        break;
                                                }//end of 
if(out==watermarkLength)
                                        }//end of for l
                                        if(finish==true){
                                                break;
                                        }//end of if(finish==true)
                                }//end of for k
                                if(finish==true){
                                        break;
                                }//end of if(finish==true)

                        }//end of for(int j=0;j<blockN;j++)
                        if(finish==true){
                                break;
                        }//end of if(finish==true)
                }//end of for(int i=0;i<blockM;i++)

        ////////}//end of if((width1==width2)&&(heigh1==heigh2))

        }//end of 
if((valueOriginal!=null)&&(valueWatermarked!=null)&&(m>0)&&(n>0))
        watermark=tempWatermark.toString();
        //watermark=watermark.concat("--->");
        return watermark;
    }//end of extract()
```

Coding Untuk Perhitungan PSNR

```
private double hitungPSNR(int[][][]image1,int[][][]image2){
              double PSNR=0;
              if((image1!=null)&&(image2!=null)){
                     int X,Y,Z;

       if(image1.length<image2.length){X=image1.length;}else{X=image2.length;}

       if(image1[0].length<image2[0].length){Y=image1[0].length;}else{Y=image2[0].
length;}

       if(image1[0][0].length<image2[0][0].length){Z=image1[0][0].length;}else{Z=i
mage2[0][0].length;}
                     double SSE=0;
                     for(int i=0;i<X;i++){
                            for(int j=0;j<Y;j++){
                                   for(int k=0;k<Z;k++){
                                          double SE=Math.pow((image1[i][j][k]-
image2[i][j][k]), 2);
                                          SSE=SSE+SE;
                                   }//end of for k
                            }//end of for j
                     }//end of for i
                     System.out.println("SSE: "+(SSE));
                     double MSE=((double)1/(double)(X*Y*Z))*SSE;
                     System.out.println("MSE: "+(MSE));
                     //PNSR=20*log(255/sqrt(MSE));
                     PSNR=20*(Math.log10((double)255/(Math.sqrt(MSE))));
              }//end of if((image1!=null)&&(image2!=null))
              return PSNR;
       }//end of  hitungPSNR()
```