# *Penggunaan Least Significant Bit pada Gambar untuk Watermarking*

Tanur Rizaldi Rahardjo (13519214)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13519214@std.stei.itb.ac.id

***Abstract*—Least significant bit, atau singkatnya LSB pada bidang citra sering kali digunakan untuk menyimpan data-data yang tersembunyi. Beberapa teknik steganografi menggunakan LSB untuk menyimpan sembarang data ke citra. Hal tersebut disebabkan oleh perubahan pada LSB tidak menyebabkan citra berubah banyak sehingga umumnya dapat mengelabuhi mata manusia. Dengan menggunakan hal tersebut, kita dapat menyisipkan gambar atau data lain terkait watermarking ke LSB sembarang gambar.**

***Keywords—least significant bit; pengolahan citra; penyisipan data***

## I. PENDAHULUAN

*Watermark* adalah salah satu alat identifikasi yang digunakan pada banyak medium. Dari *watermark* fisik seperti pada uang hingga *digital watermark* yang disisipkan pada media citra, video, dan lain-lain. Umumnya *watermark* disisipkan dengan metode yang sangat sulit untuk dimodifikasi dan duplikasi. Data identifikasi pada *watermark* dan sulitnya untuk modifikasi membuat *watermark* sering kali digunakan untuk menyulitkan untuk pembuatan *counterfeit*.

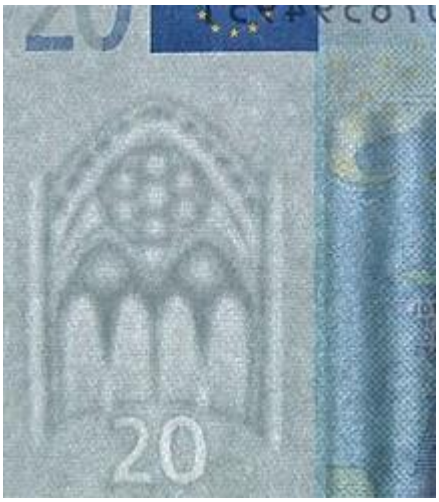
Selain menggunakan metode yang sulit untuk dimodifikasi, terkadang *watermark* dapat disisipkan dengan metode yang tersembunyi. Seperti tinta yang tidak terlihat dengan mata atau metadata yang ada pada citra. Makalah ini membahas metode *watermark* yang disisipkan pada *least significant bit* gambar. Umumnya manipulasi *least significant bit* sangat sulit untuk dideteksi oleh mata tetapi sangat mudah untuk dimodifikasi.

Metode yang dideskripsikan pada makalah ini ditujukan untuk *watermarking* sederhana dan dapat digunakan untuk identifikasi citra. Citra yang dimanipulasi umumnya pasti akan mengubah *least significant bit* citra sehingga *watermark* yang disisipkan akan berubah.

## II. LANDASAN TEORI

### A. *Watermark*

*Watermark* biasanya adalah tulisan atau citra transparan yang diletakkan diatas suatu dokumen, foto, atau media lain untuk keperluan identifikasi pemilik. Hal ini sering kali digunakan untuk membuat mempersulit pembuatan duplikasi atau *counterfeit* oleh pihak yang tidak bertanggung jawab.



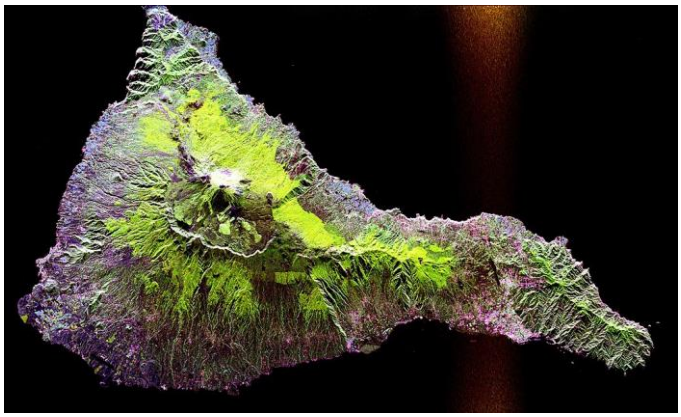

Gambar 2.1 *Watermark* denominasi 20 pada uang (Sumber: en.wikipedia.org)

Banyak metode pembuatan *watermark* yang dapat digunakan, masing-masing memiliki kelebihan dan kekurangan. Metode dengan tinta tak terlihat sering kali digunakan pada *watermark* fisik seperti uang. Metode tersebut sulit untuk dimodifikasi dan direplikasi sesuai dengan *watermark* asli. Selain itu memudahkan identifikasi dengan relatif mudahnya mengecek *watermark* dengan menggunakan sinar ultraviolet atau lainnya.

Untuk *digital watermark* metode yang sederhana adalah menuliskan informasi pada metadata media digital. Sayangnya metode ini sangat mudah untuk dimodifikasi jika digunakan secara langsung. Salah satu alat yang dapat juga digunakan untuk memperkuat *digital watermark* adalah menyisipkan *watermark* langsung pada kontennya, tetapi sedemikian rupa sehingga hasil penyisipan tidak berbeda jauh jika dibedakan dengan indra manusia.

B. Citra

Citra adalah representasi suatu benda fisik atau konsep yang umumnya dibuat dari fotografi, ilustrasi atau media lain.



Gambar 2.2 Contoh citra *Teide volcano* (Sumber: en.wikipedia.org)

Terdapat berbagai tipe citra, yakni:

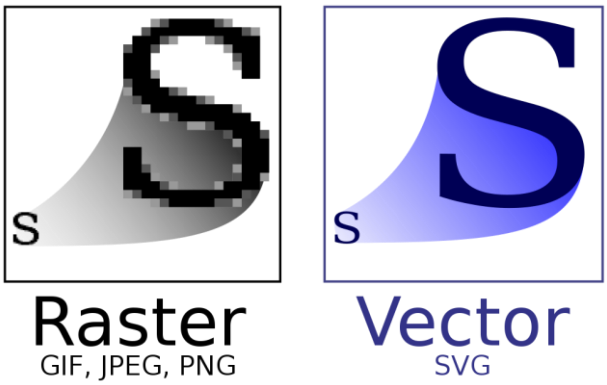
- Fotografi

  Citra fotografi ditangkap menggunakan kamera yang menangkap cahaya lingkungan menggunakan sensor. Citra fotografi biasanya ditangkap pada sensor hitam putih atau berwarna.

Gambar 2.3 Citra fotografi hitam putih (Sumber: en.wikipedia.org)

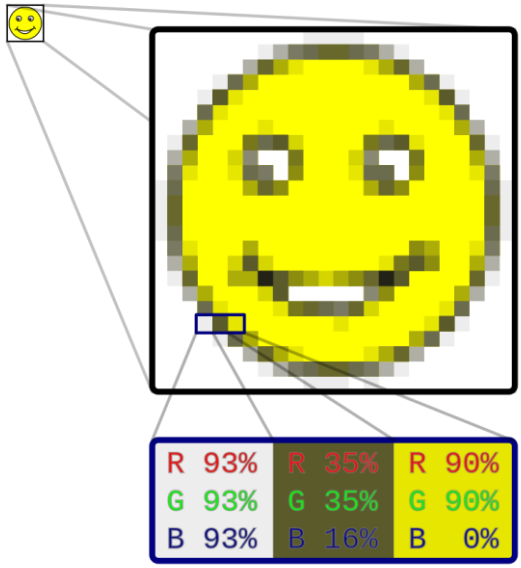
- Grafik

  Citra grafik adalah citra yang dibuat oleh *software*. Citra grafik dikategorikan menjadi *vector* dan *raster*. Umumnya citra fotografi digital diproses menjadi citra grafik *raster*. Sedangkan citra grafik *vector* umumnya adalah citra yang sepenuhnya dibuat pada *vector graphics software*. Kelebihan *vector* adalah *perfect scaling* dengan definisi citra dibuat oleh persamaan garis dan kurva.




Gambar 2.4 Citra grafik *vector* (Sumber: en.wikipedia.org)

Kelebihan *raster* adalah mudahnya untuk memanipulasi gambar dengan *pixel* jika dibandingkan *vector* dan tingginya resolusi membuat kelebihan *vector* kurang relevan pada banyak kasus.




Gambar 2.5 Citra grafik *raster* (Sumber: en.wikipedia.org)

- Ilustrasi

  Citra ilustrasi adalah citra yang dibuat oleh tangan atau *software* komputer. Umumnya dibuat untuk merepresentasikan konsep atau sebuah ide. Banyak *style* yang ada pada citra ilustrasi, seperti realistik, kartun, atau abstrak.

Gambar 2.6 Citra illustrasi (Sumber: en.wikipedia.org)

## III. IMPLEMENTASI

Berikut adalah kode implementasi dari watermark

```
classdef IF4073_Plate_Detection < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public) ...

    properties (Access = private)
        OriginalImg
        GrayImg
        CroppedImg
    end
    methods (Access = private)
        function Execute(app, I, W)
            app.OriginalImg = I;
            app.GrayImg = imbinarize(W(:, :, 3));

            res = zeros(size(I));
            for col = 1:size(I, 3)
                for r = 1:size(I, 1)
                    for c = 1:size(I, 2)
                        if r < size(W, 1) && c < size(W, 2)
                            res(r, c, col) = bitand(I(r,c,col), 0xFE) + bitand(uint8(app.GrayImg(r,c)), 1);
                        else
                            res(r, c, col) = bitand(I(r,c,col), 0xFE);
                        end
                    end
                end
            end
            app.ImageCrop.ImageSource = uint8(res);

            b = bitand(uint8(res), 0x1);
            for col = 1:size(b, 3)
                for r = 1:size(b, 1)
                    for c = 1:size(b, 2)
```

Gambar 3.1

```
                    b = bitand(uint8(res), 0x1);
                    for col = 1:size(b, 3)
                        for r = 1:size(b, 1)
                            for c = 1:size(b, 2)
                                if b(r, c, col) > 0
                                    b(r, c, col) = 256;
                                end
                            end
                        end
                    end
                    figure, imshow(uint8(b));

                    imwrite(uint8(res), "test.jpg")
                end
            end

            % Callbacks that handle component events
            methods (Access = private)

                % Code that executes after component creation
                function startupFcn(app)

                end

                % Button pushed function: BrowseforImageButton
                function BrowseforImageButtonPushed(app, event)
                    [file,path] = uigetfile({'*.png;*.jpg;*.jpeg','Images'});
                    if (file ~= 0)
                        app.OriginalImg = imread(fullfile(path,file));
                        app.ImageSrc.ImageSource = app.OriginalImg;
                    end
                end
```

Gambar 3.2

```
        end

        % Callback function: not associated with a component
        function PlateTemplateDropDownValueChanged(app, event)

        end

        % Callback function: not associated with a component
        function RotateImageCheckBoxValueChanged(app, event)

        end

        % Button pushed function: ExecuteButton
        function ExecuteButtonPushed(app, event)
            app.Execute(app.ImageSrc.ImageSource, app.ImageGray.ImageSource);
        end

        % Button pushed function: BrowseforWatermarkButton
        function BrowseforWatermarkButtonPushed(app, event)
            [file,path] = uigetfile({'*.png;*.jpg;*.jpeg','Images'});
            if (file ~= 0)
                app.GrayImg = imread(fullfile(path,file));
                app.ImageGray.ImageSource = app.GrayImg;
            end
        end
    end

    % Component initialization
    methods (Access = private)
```

Gambar 3.3

```
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 753 586];
            app.UIFigure.Name = 'MATLAB App';

            % Create ImagesPanel
            app.ImagesPanel = uipanel(app.UIFigure);
            app.ImagesPanel.Title = 'Images';
            app.ImagesPanel.Position = [39 40 682 506];

            % Create SourceLabel
            app.SourceLabel = uilabel(app.ImagesPanel);
            app.SourceLabel.Position = [105 246 44 22];
            app.SourceLabel.Text = 'Source';

            % Create EmbeddedLabel
            app.EmbeddedLabel = uilabel(app.ImagesPanel);
            app.EmbeddedLabel.Position = [512 246 63 22];
            app.EmbeddedLabel.Text = 'Embedded';

            % Create WatermarkLabel
            app.WatermarkLabel = uilabel(app.ImagesPanel);
            app.WatermarkLabel.Position = [302 245 63 22];
            app.WatermarkLabel.Text = 'Watermark';
```

Gambar 3.4

```
            app.WatermarkLabel.Text = 'Watermark';

            % Create ImageCrop
            app.ImageCrop = uiimage(app.ImagesPanel);
            app.ImageCrop.Position = [448 266 189 205];

            % Create ImageGray
            app.ImageGray = uiimage(app.ImagesPanel);
            app.ImageGray.Position = [237 266 189 205];

            % Create ImageSrc
            app.ImageSrc = uiimage(app.ImagesPanel);
            app.ImageSrc.Position = [31 266 189 205];

            % Create BrowseforImageButton
            app.BrowseforImageButton = uibutton(app.ImagesPanel, 'push');
            app.BrowseforImageButton.ButtonPushedFcn = createCallbackFcn(app, @BrowseforImageButtonPushed, true);
            app.BrowseforImageButton.Position = [235 175 195 33];
            app.BrowseforImageButton.Text = 'Browse for Image ...';

            % Create ExecuteButton
            app.ExecuteButton = uibutton(app.ImagesPanel, 'push');
            app.ExecuteButton.ButtonPushedFcn = createCallbackFcn(app, @ExecuteButtonPushed, true);
            app.ExecuteButton.Position = [282 58 100 23];
            app.ExecuteButton.Text = 'Execute';

            % Create BrowseforWatermarkButton
            app.BrowseforWatermarkButton = uibutton(app.ImagesPanel, 'push');
            app.BrowseforWatermarkButton.ButtonPushedFcn = createCallbackFcn(app, @BrowseforWatermarkButtonPushed, true);
            app.BrowseforWatermarkButton.Position = [236 123 195 33];
            app.BrowseforWatermarkButton.Text = 'Browse for Watermark ...';
```

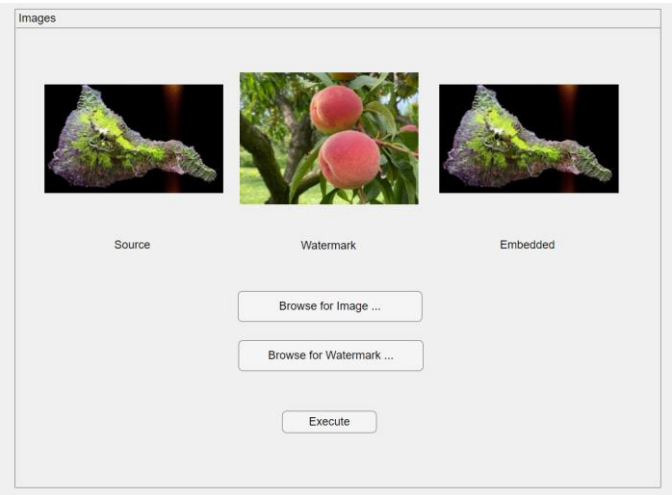
Gambar 3.5

```
            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = IF4073_Plate_Detection

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            % Execute the startup function
            runStartupFcn(app, @startupFcn)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

Gambar 3.6

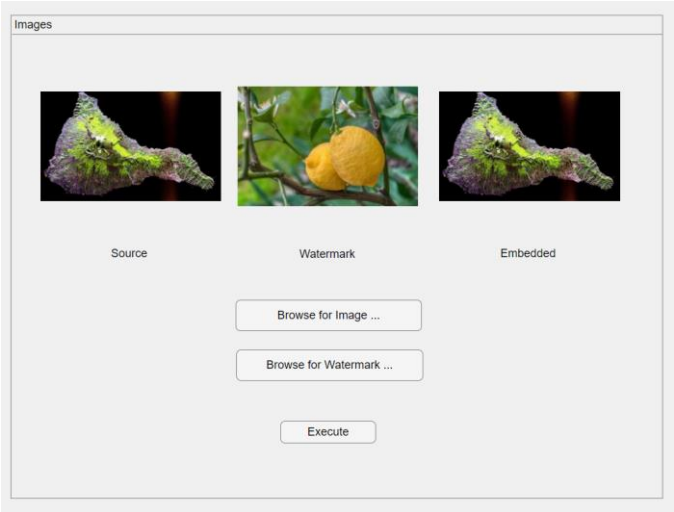
## IV. HASIL & ANALISIS

Berikut adalah hasil eksekusi program pada beberapa gambar



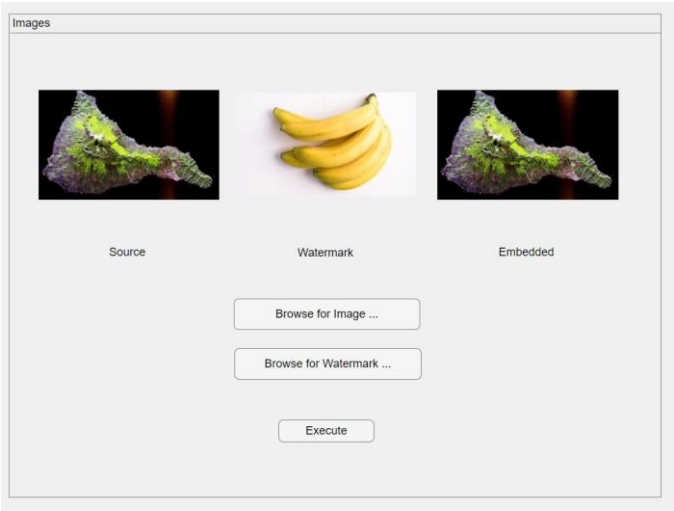

Gambar 4.1

Gambar 4.2 Bit plane LSB



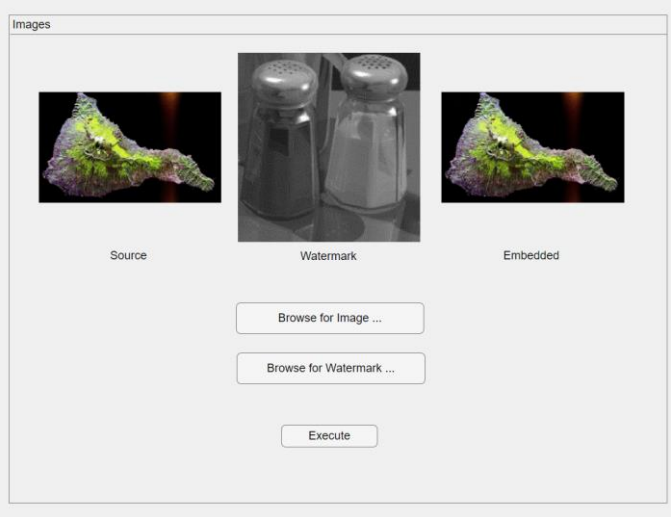

Gambar 4.3

Gambar 4.4 Bit plane LSB

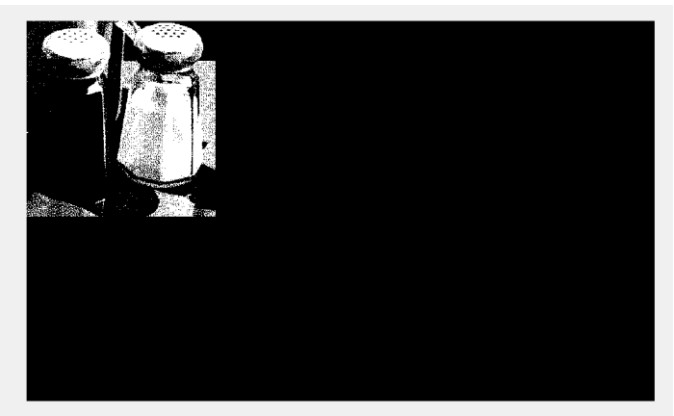



Gambar 4.5

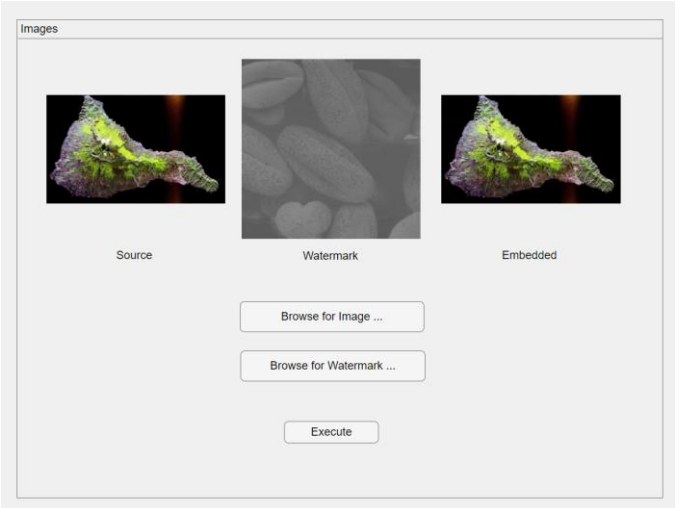
Gambar 4.6 Bit plane LSB




Gambar 4.7



Gambar 4.8 Bit plane LSB




Gambar 4.9

Gambar 4.10 Bit plane LSB

## V. HASIL & ANALISIS

Dari hasil yang dihasilkan dapat dilihat bahwa hasil *watermark* terlihat cukup jelas jika dicek dengan aplikasi atau program yang sesuai. Karena relatif sederhananya metode ini, metode ini sangat mudah dimodifikasi jika diketahui adanya *watermark*.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Kode dapat diakses pada
https://github.com/Lock1/IF4073-Makalah

Bandung, 19 Desember 2022
Tanur Rizaldi Rahardjo / 13519214