MODUL I
REPRESENTASI PENGOLAHAN CITRA PADA PEMROGRAMAN WEB

I. TUJUAN

1. Praktikan mengenal lingkungan pemograman WEB.
2. Praktikan mengerti cara mereprsentasikan citra pada HTML, menerima data dalam bentuk array, mengubah nilai RGB.
3. Praktikan mengetahui histogram citra pada pemrograman Web

II. TEORI

2.1 HTML 5 Canvas

Elemen <canvas> digunakan untuk menggambar grafik dengan cepat melalui scripting (biasanya Javascript). Elemen <canvas> hanya digunakan sebagai wadah untuk grafis. Anda harus menggunakan script untuk menggambar grafik. Canvas mempunyai beberapa metode untuk *drawing paths*, *boxes*, *circle*, text, dan *adding images*.

Untuk mempermudah proses pengelolaan citra, maka digunakan pc.js.

Sintaks fungsi utama pc.js dalam manipulasi citra.

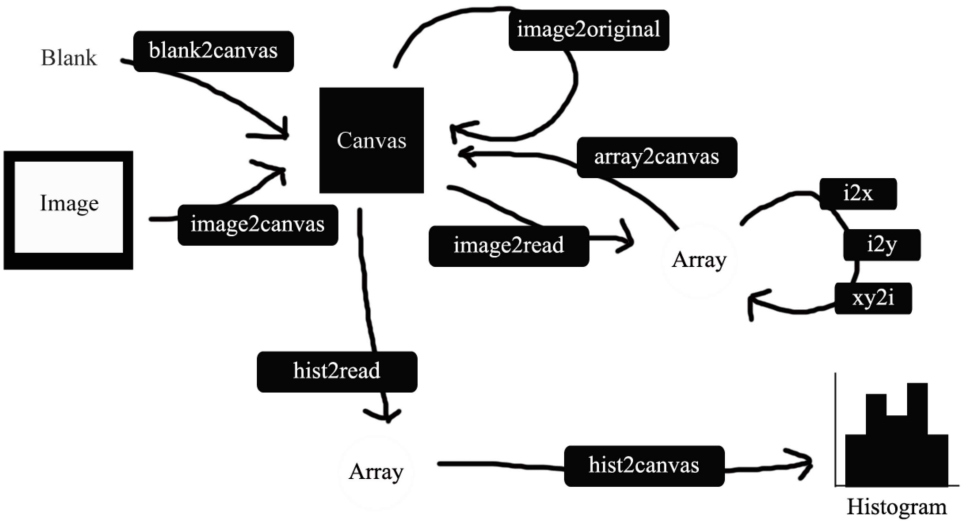
```
function pc(canvas){
	this.canvas = canvas
	this.context = this.canvas.getContext('2d')
	this.width = 0
	this.height = 0
	this.imgsrc = ""
	this.image2read = function(){
		this.originalLakeImageData = this.context.getImageData(0,0,
		this.width, this.height)
		this.resultArr = new Array()
		this.tempArr = new Array()
		this.tempCount = 0
		for(var i=0; i<this.originalLakeImageData.data.length; i++){
			this.tempCount++
			this.tempArr.push(this.originalLakeImageData.data[i])
			if(this.tempCount == 4){
				this.resultArr.push(this.tempArr)
				this.tempArr = []
				this.tempCount = 0
			}
		}
		info('image2read Success ('+this.imgsrc+') :
		'+this.width+'x'+this.height)
		return this.resultArr
	}

	this.blank2canvas = function(w,h){
		this.width = w
		this.height = h
		this.canvas.width = this.width
		this.canvas.height = this.height
		this.imgsrc = "Blank"
		info('blank2canvas Success (Blank '+w+'x'+h+')')
```

```
}

this.image2canvas = function(imgsrc){
	var imageObj = new Image()
	var parent = this
	imageObj.onload = function() {
		parent.canvas.width = imageObj.width
		parent.canvas.height = imageObj.height
		parent.context.drawImage(imageObj, 0, 0)
		parent.width = imageObj.width
		parent.height = imageObj.height
		info('image2canvas Success ('+imgsrc+')')
	}
	imageObj.src = imgsrc
	this.imgsrc = imgsrc
}

this.image2original = function(){
	if(this.imgsrc == ""){
		error("image2original Failed : Image Source not found!")
	}
	else if(this.imgsrc == "blank"){
		this.blank2canvas(this.width, this.height)
		info("image2original Success")
	}
	else{
		this.image2canvas(this.imgsrc)
		info("image2original Success")
	}
}

this.array2canvas = function(arr){
	this.imageData = this.context.getImageData(0,0, this.width,
	this.height)
	if(this.imageData.data.length != arr.length*4){
		error("array2canvas Failed to Execute")
		return false
	}
	for(var i = 0; i < arr.length; i++){
		this.imageData.data[(i*4)] = arr[i][0]
		this.imageData.data[(i*4)+1] = arr[i][1]
		this.imageData.data[(i*4)+2] = arr[i][2]
		this.imageData.data[(i*4)+3] = arr[i][3]
	}
	this.context.clearRect(0, 0, this.width, this.height)
	this.context.putImageData(this.imageData, 0, 0)
	info('Array2Canvas Success ('+this.imgsrc+')')
}

this.hist2read = function(arr){
	//checking
	for(var i=0; i<arr.length; i++){
		if(arr[i] < 0 || arr[i] > 3){
			error('hist2read Failed : Wrong
			parameter('+arr[i]+') : ('+this.imgsrc+')')
			return false
		}
	}
	//end of checking
	var read = this.image2read()
	var resArr = new Array()
	for(var i=0; i<arr.length; i++){
		var tempArr = new Array(read.length)
```

```
                for(var c=0; c<read.length; c++){
                        tempArr[c] = read[c][arr[i]]
                }
                var tempArr = tempArr.sort()
                var fixedval = new Array(256)
                for(var init=0; init<256; init++) fixedval[init] = 0

                for(var a = 0; a< tempArr.length; a++){
                        fixedval[tempArr[a]]++
                }
                resArr.push(fixedval)
        }
        return resArr
}

this.hist2canvas = function(arr,fontsize){
        if(arr == undefined){
                error("hist2canvas Failed to execute")
                return false
        }
        var wc = this.width
        var hc = this.height
        var max = Math.max.apply(null,arr)
        var gmax = max - (max % 100) + 100
        var gmid = Math.ceil(gmax/2)
        var context = this.context
        var margin = 5
        context.clearRect(0, 0, this.width, this.height)

        context.fillStyle = "#f0f0f0";
        context.fillRect(0,0,wc,hc);

        context.font= fontsize+"px Arial";
        var txt1= gmax
        var txt2 = gmid

        widthfontmax = context.measureText(txt1).width
        widthfontmid = context.measureText(txt2).width

        context.fillStyle = "#000000";
        context.fillText(txt1,margin,fontsize);

        context.beginPath();

        context.moveTo(widthfontmax+margin*2,0);
        context.lineTo(widthfontmax+margin*2,hc);

        context.moveTo(0,(hc - margin*2));
        context.lineTo(wc,(hc - margin*2));

        if(gmid == gmax){
                gmid = -1
                txt2 = ""
        }
        else{
                context.fillStyle = "#000000";
                context.fillText(txt2,margin,fontsize + ((hc/2) –
                margin*2));
        }

        var marginbottom = (margin*2)
        var histheight = hc - marginbottom - (widthfontmax/2)
        var histwidth = wc - 2*margin - widthfontmax - 1
```

```
                for(var i=0; i<arr.length; i++){
                        context.moveTo(2*margin + widthfontmax + 1 + (((i+1)/256)*histwidth),hc - ((arr[i]/gmax)*histheight)
- marginbottom)
                        context.lineTo(2*margin + widthfontmax + 1 + (((i+1)/256)*histwidth),(hc - marginbottom));
                }

                context.strokeStyle = '#000000';
                context.stroke();

                var grad = context.createLinearGradient(2*margin + widthfontmax + 1,hc,wc,hc);
                grad.addColorStop(0, '#000000');
                grad.addColorStop(1, '#ffffff');

                context.fillStyle = grad;
                context.fillRect(2*margin + widthfontmax + 1,hc - marginbottom ,wc,hc);

                info("Hist2canvas Success")
        }

        this.i2x = function(i){
                return (i % this.width)
        }

        this.i2y = function(i){
                return ((i - (i % this.width))/ this.width)
        }

        this.xy2i = function(x,y){
                return (y * this.width) + (x)
        }

}
```

Gambaran Penggunaan pc.js



Keterangan Gambar :

Nama fungsi	Deskripsi	Sintaks
blank2canvas	Membuat image kosong ke canvas	blank2canvas(width,height)
image2canvas	Memasukkan image ke canvas	image2canvas(url)
image2original	Mengembalikan citra canvas ke keadaan setelah proses blank2canvas atau image2canvas	image2original()
image2read	Membaca image dan mengembalikannya dalam bentuk array yang sudah dimodifikasi (dalam hitungan per pixel untuk setiap index array)	image2read()
hist2read	Membaca image dan mengembalikan dalam bentuk array (data dalam array sudah disorting secara ascending)	hist2read(array[0..3])
hist2canvas	Mengubah data array histogram (dari proses hist2read) ke dalam bentuk citra (ke canvas)	hist2canvas(array, fontsize)
array2canvas	Mengubah data array ke dalam bentuk citra (ke canvas)	array2canvas()
i2x	Mengubah nilai index array ke koordinat x	i2x(index)
i2y	Mengubah nilai index array ke koordinat y	i2y(index)
xy2i	Mengubah nilai x dan y menjadi index array	xy2i(x,y)

Contoh 1 :

Harus dijalankan di web server, contoh : Localhost

modul_1.html

```
<!DOCTYPE HTML>
<html>
<head>
<title>Modul 1 Contoh 1</title>
</head>
<body>
  <canvas id='canvas1'></canvas></div>
  <canvas id='canvas2'></canvas></div>
  <hr>
  <button id='read'>READ IMAGE</button></div>
  <hr>
  <button id='ori'>ORIGINAL IMAGE</button></div>
  <hr>
  <select id='histval'>
        <option value='0'>Red</option>
        <option value='1'>Green</option>
```

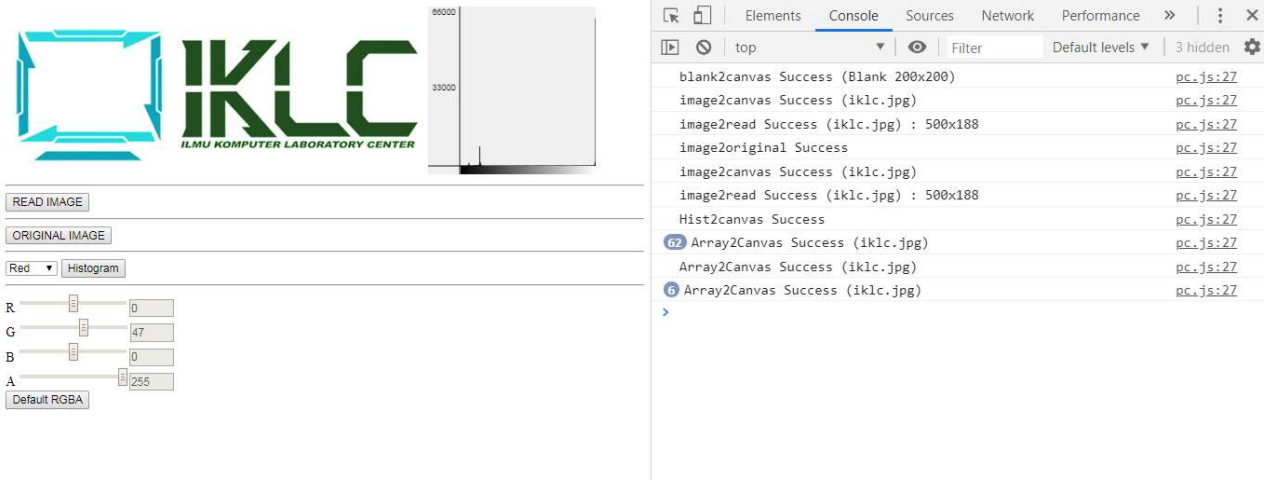
```
        <option value='2'>Blue</option>
  </select>
  <button id='hist1'>Histogram</button>
  <hr>
  R <input type='range' min='-255' max='255' data-id='1' id='ch1'><input type='text' id='chv1' size='3' disabled value='0'><br>
  G <input type='range' min='-255' max='255' data-id='2' id='ch2'><input type='text' id='chv2' size='3' disabled value='0'><br>
  B <input type='range' min='-255' max='255' data-id='3' id='ch3'><input type='text' id='chv3' size='3' disabled value='0'><br>
  A <input type='range' min='0' max='255' data-id='4' id='ch4' value='255'><input type='text' id='chv4' size='3' disabled
value='255'><br>
  <button id='default'>Default RGBA</button>
</body>
<style>
body{
  background : rgba(255,255,255,1);
}
</style>
<script src='pc.js'></script>
<script>
  var canvas = document.getElementById('canvas1')
  var obj = new pc(canvas)
  obj.image2canvas("iklc.jpg")

  var canvas2 = document.getElementById('canvas2')
  var obj2 = new pc(canvas2)
  obj2.blank2canvas(200,200)

  var tes = new Array()
  document.getElementById('read').addEventListener('click',function(){
        tes = obj.image2read()
  })

  document.getElementById('ori').addEventListener('click',function(){
        obj.image2original()
  })

  function rgbachange(){
        //copy array to array without reference
        tesbackup = new Array()
        for(var c=0; c<tes.length; c++){
              temp = new Array()
              for(var d=0; d<4; d++){
                    temp.push(tes[c][d])
              }
              tesbackup.push(temp)
        }
        //end of copy
        for(var j=0; j<tesbackup.length; j++){
                    tesbackup[j][0] += parseInt(document.getElementById('ch1').value)
                    tesbackup[j][1] += parseInt(document.getElementById('ch2').value)
                    tesbackup[j][2] += parseInt(document.getElementById('ch3').value)
                    tesbackup[j][3] = parseInt(document.getElementById('ch4').value)
        }
        for(var i=1; i<=4;i++) document.getElementById('chv'+i).value = document.getElementById('ch'+i).value
        obj.array2canvas(tesbackup)
  }

  for(var i=1; i<=4;i++){
        document.getElementById('ch'+i).addEventListener('input',function(){
              rgbachange()
        })
  }

  document.getElementById('default').addEventListener('click',function(){
```

```
        for(var i=1; i<=3;i++){
                document.getElementById('ch'+i).value = 0
        }
        document.getElementById('ch4').value = 255
        rgbachange()
  })

  document.getElementById('hist1').addEventListener('click',function(){
        var hist = obj.hist2read([parseInt(document.getElementById("histval").value)]) //Call [R,G,B,A] as [0,1,2,3], can input more than 1 but only [0..3] parameter only
        obj2.hist2canvas(hist[0],10)
  })
</script>
```

Keterangan tambahan :

1. Untuk melihat hasil, gunakan shortcut Ctrl + Shift + J (pada Google Chrome) untuk membuka Developer Console
2. Untuk menggunakan fungsi - fungsi utama pada pc.js, harus dibuat objek terlebih dahulu, contoh : var obj = new pc(canvas)
3. Array yang dihasilkan dari image2read □ array[[r,g,b,a], [r,g,b,a], …] dengan
   r : red (0 - 255)
   g : green (0 - 255)
   b : blue (0 - 255)
   a : alpha (0 - 255)
4. Parameter dalam hist2read harus dalam bentuk array
   Contoh :
   hist2read([0,1,2]), jumlah array bebas, tetapi hanya terbatas untuk nilai 0 sampai 3 dimana (0,1,2,3) adalah (r,g,b,a), array yang dihasilkan adalah nilai – nilai dari channel yang dipilih dan terurut secara ascending

Hasil modul_1.html



MODUL II
PEMROGRAMAN IMAGE I

I. TUJUAN

1. Praktikan dapat memahami tentang mengubah nilai Brightness, Grayscale, Threshold, dan negative Image pada pemrograman Web.

II.TEORI

2.1 Brightness

Brightness adalah metoda untuk mempercerah citra dengan cara menambah seluruh derajat keabuan citra dengan bilangan tertentu.

2.2 Grayscale

Citra greyscale merupakan suatu citra yang mempunyai derajat keabuan 8 bit. Matriks yang digunakan hanya 2D (1 layer). Karena citra berwarna terdiri dari 3 layer RGB, maka untuk mendapatkan citra greyscale, bias saja dengan cara mengambil salah satu layer atau mencari rata-rata dari setiap elemen dalam tiap layer.

2.3 Threshold

Threshold adalah suatu cara mempertegas citra dengan mengubah citra menjadi hitam dan putih (2 bit). Dalam threshold ini, harus ditetapkan suatu variable yang berfungsi sebagai batas untuk mengkonversi elemen matriks citra menjadi hitam atau putih. Jika nilai elemen matriks dibawah nilai ini, dikonversi menjadi nilai 0 (hitam), jika diatas nilai ini, elemen dikonversi menjadi 1. Untuk citra RGB, terlebih dahulu diubah menjadi greyscale.

2.4 Negative Image

Citra negative merupakan representasi kebalikan citra asli. Karena nilai maksimum matriks adalah 255, maka untuk memperoleh kebalikan elemen, didapat dengan mengurangi elemen tersebut dari 255.

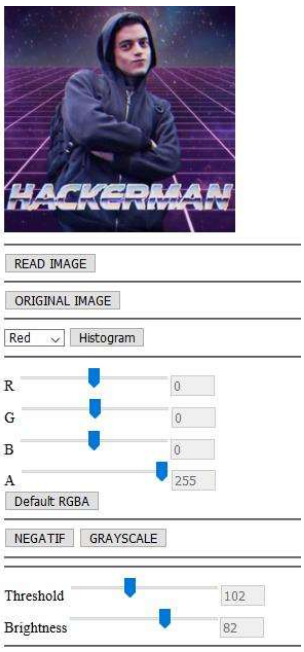
Image.html

```
document.getElementById('negatif').addEventListener('click',function(){
        //copy array to array without reference
        tesbackup = new Array()
        for(var c=0;c<tes.length;c++){
          temp = new Array()
          for(var d=0;d<4;d++){
                    temp.push(tes[c][d])
          }
          tesbackup.push(temp)
        }
        //end of copy
        for(var i=0;i<tesbackup.length;i++){
          tes[i][0] = (255-tesbackup[i][0])
          tes[i][1] = (255-tesbackup[i][1])
```

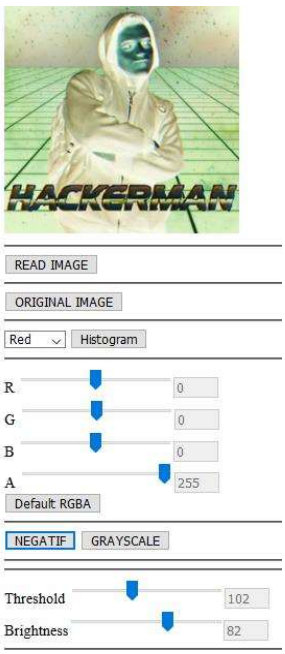
```
    tes[i][2] = (255-tesbackup[i][2])
    tes[i][3] = tesbackup[i][3]
  }
  obj.array2canvas(tes)
  })
  document.getElementById('grayscl').addEventListener('click',function(){
    //copy array to array without reference
    tesbackup = new Array()
    for(var c=0;c<tes.length;c++){
    temp = new Array()
    for(var d=0;d<4;d++){
            temp.push(tes[c][d])
    }
    tesbackup.push(temp)
  }
  //end of copy
  for(var i=0;i<tesbackup.length;i++){
    var total = Math.floor((tesbackup[i][0] + tesbackup[i][1] + tesbackup[i][2])/3)
    tes[i][0] = total
    tes[i][1] = total
    tes[i][2] = total
    tes[i][3] = tesbackup[i][3]
  }
  obj.array2canvas(tes)
  })
  document.getElementById('threshold').addEventListener('input',function(){
    document.getElementById('threshold_val').value=this.value
    batas = parseInt(this.value)
    //copy array to array without reference
    tesbackup = new Array()
    for(var c=0;c<tes.length;c++){
    temp = new Array()
    for(var d=0;d<4;d++){
            temp.push(tes[c][d])
    }
    tesbackup.push(temp)
  }
  //end of copy
  for (var i=0;i<tes.length;i++){
    gabung = Math.floor((tesbackup[i][0] + tesbackup[i][1] + tesbackup[i][2])/3)
    if(gabung<batas){
            gabung=0
    }
    else{
            gabung=255
    }
    tes[i][0] = gabung
    tes[i][1] = gabung

    tes[i][2] = gabung
    tes[i][3] = tes[i][3]
  }
  obj.array2canvas(tesbackup)
  })

  document.getElementById('brightness').addEventListener('input',function(){
    document.getElementById('brightness_val').value=this.value
    p = parseInt(this.value)
    //copy array to array without reference
    tesbackup = new Array()
    for(var c=0;c<tes.length;c++){
    temp = new Array()
    for(var d=0;d<4;d++){
```

```
                temp.push(tes[c][d])
            }
            tesbackup.push(temp)
        }
        //end of copy
        for (var i=0;i<tes.length;i++){

            tesbackup[i][0] = (tes[i][0]+p)
            tesbackup[i][1] = (tes[i][1]+p)
            tesbackup[i][2] = (tes[i][2]+p)
            tesbackup[i][3]= tes[i][3]
            }
            obj.array2canvas(tesbackup)

            })
```

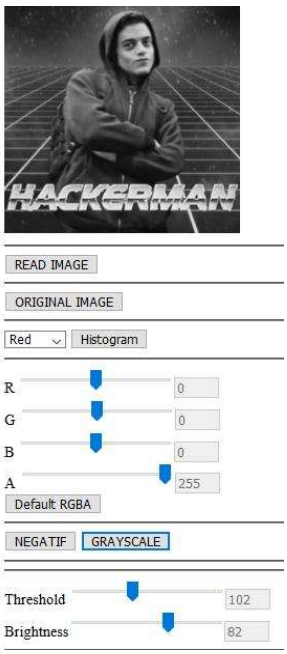
Hasil :



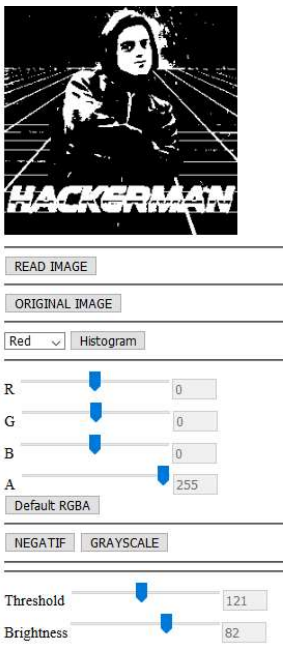
Gambar 2.1
Gambar Original



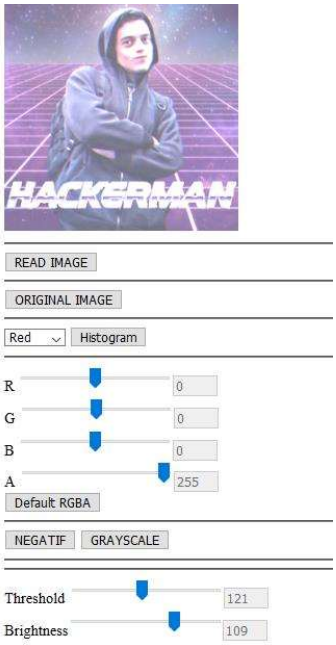
Gambar 2.2
Negative Image



Gambar 2.3
Grayscale Image



Gambar 2.4
Gambar Treshold



Gambar 2.5
Dengan filter brightness