PERANCANGAN SISTEM *SCRATCHREMOVAL* DAN *DEBLURRING* TERHADAP CITRA *QR-CODE*

Ricco Felixon[1)],**Tony**[2)],**Dyah E. Herwindiati**[3)]

[1,2,3)] Program StudiTeknik Informatika
Fakultas Teknologi Informasi, Universitas Tarumanagara
Jl. Let. Jend. S. Parman No. 1, Jakarta 11440 Indonesia
Email:[1)]crimesoncrimes@gmail.com,[2)]Dyahh@fti.untar.ac.id, [3)]tony.b@fti.untar.ac.id

ABSTRACT

QR-CODE is the 2-D code that can be stored 7089 characters, can be read very quick, and can be read at 360 degree angle. Unfortunately the advantages of QR-CODE can be reduced by a noise like scratch and blur. The system that will be created is a software application with an input image of the QR-CODE image that has been generated through the website https://www.the-qrcode-generator.com and is given a scratch, blur, a combination of blur and scratch, and the combination of scratch and blur. The methods used for data scratch is scratch removal and a deblurring method used for data blur is Enchanted Iterative Van Cittert. On the test results obtained average estimation system error of scratch removal, deblurring, deblurring scratch removal, and scratch removal deblurring of 1.3354, 1.7560 2.5217%,%, and% 5.8432. The result of the QR-CODE program tested with QRCODE READER obtained average estimation 98% of the QR-CODE result can be read properly.

Key words

QR-CODE, scratch, blur, scratch removal, deblurring, Enchanted Itterative Van Citert

1. Pendahuluan

Di era globalisasi yang terus mengalami perubahan dan perkembangan terhadap Teknologi Informasi telah banyak mempengaruhi sendi-sendi kehidupan masyarakat. Keadaan ini terbukti dengan semakin berkembangnya teknologi informasi dalam bidang bisnis, kesehatan, pendidikan, sosial, dan komunikasi[1].Teknologi pemrosesan dan pengolahan data berlomba-lomba untuk mengolah data dengan baik dan cepat. Salah satu contoh sarana teknologi informasi yang digunakan untuk membantu meningkatkan kecepatan pemrosesan adalah dengan menggunakan *barcode*[2].

Barcode adalah informasi terbacakan mesin (*machine readable*) dalam format visual yang tercetak. *Barcode* dibaca dengan menggunakan sebuah alat baca *barcode* atau lebih dikenal dengan *barcode scanner*. Seiring semakin bertambahnya penggunaan *barcode*, kini barcode tidak hanya bisa mewakili karakter angka saja tapi sudah meliputi seluruh kode ASCII. Kebutuhan akan kombinasi kode yang lebih rumit itulah yang kemudian melahirkan inovasi baru berupa kode matriks dua dimensi yang berupa kombinasi kode matriks bujur sangkar. 2 dimension *barcode* ini diantaranya adalah *PDF Code*, *QR-CODE*, dan *Matrix Code*[3].

QR-CODE merupakan singkatan dari Quick Response Code, atau dapat diterjemahkan menjadi kode respon cepat. *QR-CODE* dikembangkan oleh Denso Corporation, sebuah perusahaan Jepang yang banyak bergerak di bidang otomotif. *QR-CODE* ini dipublikasikan pada tahun 1994 dengan tujuan untuk pelacakan kendaraan di bagian manufaktur dengan cepat dan mendapatkan respon dengan cepat[4].

QR-CODE juga tidak luput dari kekurangan. Ada pun beberapa kekurangan dari *QR-CODE* antara lain :

1. Pengguna harus dilengkapi dengan kamera ponsel dan perangkat lunak pemindai (*QR-CODEscanner*) yang benar untuk dapat memindai citra *QR-CODE*
2. Saat hanya *smartphone* yang secara teknis dapat melakukan pemindaian
3. Proses pemindaian sangat tergantung pada kualitas kamera [5]

Berbagai kondisi fisik *QR-CODE* menjadi pengaruh yang sangat besar dalam proses pemindaian. Adapun beberapa kondisi yang menyebabkan pemindaian tidak berjalan dengan baik yaitu: *scratch*(gores) dan *blur*(kekaburan).

Untuk mengatasi kekurangan pada proses pemindaian *QR-CODE* maka dalam topik skripsi ini proses pemindaian *QR-CODE*akan disandingkan dengan proses pengolahan citra sebelum dilakukan proses pemindaian.

Pengolahan citra yang dilakukan terhadap citra *QR-CODE* yaitu *deblurring* dan *scratch removal*.

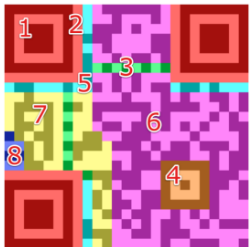
2. Landasan Teori

2.1 *QR-CODE*

QR-CODE merupakan teknik yang mengubah data tertulis menjadi kode-kode 2 dimensi yang tercetak kedalam suatu media yang lebih ringkas. *QR-CODE* juga dapat menyimpan hingga 7089 karakter data numerik, 4296 karakter data alfanumerik, 2953 byte data biner, dan 1.817 Kanji atau Kana. *QR-CODE* memiliki struktur yang unik sehingga dapat dibaca 360 derajat dan memiliki error correction yang memungkinkan *QR-CODE* untuk tetap dapat terbaca jika terkena kotor [6].

2.2 Struktur *QR-CODE*

Struktur *QR-CODE*akan dijelaskan pada Gambar 1.



Gambar 1Struktur *QR-CODE*

Sumber:Peter Kieseberg, QR Code Security, https://www.sba-research.org/wp-content/uploads/publications/QR_Code_Security.pdf, 24 Februari 2015

Keterangan :

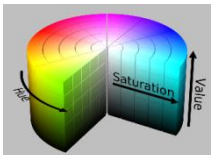
1. *Finder Pattern* (1): terdiri dari tiga struktur yang identik dan berlokasi di seluruh sudut *QR-CODE* kecuali pola berbentuk kotak yang berukuran lebih kecil bernomor empat. Setiap pola didasarkan pada sebuah kotak hitam dikelilingi oleh matriks dari modul modul putih yang dikelilingi oleh modul hitam lagi. *Finder Pattern* ini memungkinkan *decoder* perangkat lunak untuk memindai *QR-CODE* dan menentukan orientasi yang benar.
2. *Separators* (2): *separators* ditandai dengan nomor 2 memiliki warna putih dan memiliki lebar lebih kurang satu piksel serta meningkatkan *recognizability* dari *Finder Pattern* karena *separators* memisahkan *Finder Pattern* dari data aktual.
3. *Timing Patterns* (3): modul hitam dan putih pada Timing Patterns memungkinkan decoder perangkat lunak untuk menentukan lebar satuan modul.
4. *Alignment Patterns* (4): Alignment Patterns membantu decoder perangkat lunak dalam perbesaran citra QR-CODE. QR-CODE versi 1 tidak Alignment Patterns. Seiring dengan bertambah besarnya versi/ukuran QR-CODE maka Alignment Patterns akan semakin ditambahkan.
5. *Format Information* (5): Pada bagian *FormatInformation* ini pembentukan terdiri dari 15 bits berada tepat disamping separators dan menyimpan informasi tentang tingkat kesalahan koreksi *QR-CODE*.
6. *Data* (6): Pada bagian *Data* dikonversikan menjadi *bit stream* dan di simpan dalam partikel 8 bits (disebut *codewords*).
7. *Error Correction* (7): Mirip seperti pada bagian *Data*, *ErrorCorrection* disimpan dalam 8 bit long *codewords*.
8. *Remainder Bits* (8): Bagian ini terdapat bit kosong bila pada bagian *Data* dan *ErrorCorrection* tidak dapat dibagi menjadi 8 bit *Codewords*.[7]

2.3 *Scratch Removal*

Proses *scratch removal* dibagi menjadi beberapa tahapan antara lain :

2.3.1 Hue Saturation Value

HSV merupakan proyeksi dari model *RGB*(*RedGreenBlue*) ke sudut *chromanon-linear*, persentase saturasi radial, dan nilai luminasi. Secara lebih detail *value* di definisikan sebagai nilai rata-rata atau maksimum dari nilai warna, *saturation* didefinisikan sebagai jarak dari diagonal, dan *hue* di definisikan sebagai arah dari warna.[8] Contoh gambar HSV akan disajikan pada Gambar 2



Gambar 2*HSVcylinder*

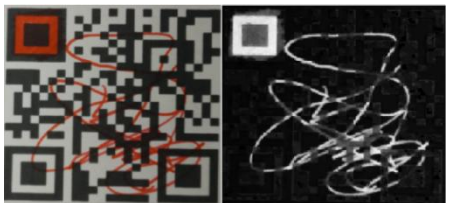
Sumber : PyImageSearch, The complete guide to building an image search engine with Python and OpenCV, http://www.pyimagesearch.com/2014/12/01/complete-guide-building-image-search-engine-python-opencv/, 25 Februari 2015

Model *HSV* dapat memisahkan komponen intensitas dari citra warna, sehingga model ini merupakan model yang ideal untuk mengembangkan algoritma pemrosesan citra. Untuk memisahkan *scratch* pada citra *QR-CODE* input akan dilakukan konversi citra menjadi *HSV*. Selanjutnya akan dipilih *layersaturation* karena pada *layer* ini warna *scratch* pada *QR-CODEinput* akan dapat ditentukan. Berikut ini rumus untuk mendapatkan *layersaturation*:

$$Saturation = 1 - \frac{3}{R+G+B}[\min(Red, Green, Blue)] \quad (1)$$

Keterangan :

R = Unsur merah dalam model *RGB*
G = Unsur hijau dalam model *RGB*
B = Unsur biru dalam model *RGB*
Saturation= Saturation menyatakan seberapa dalam warna tersebut putih di dalamnya)
Min= Nilai minimum diantara *RGB*



Gambar 3Citra input (kiri) dan layer saturation (kanan)

Sumber : Dokumentasi pribadi, 24 Februari 2015

2.3.2 *Tresholding*

Thresholding adalah suatu proses yang digunakan untuk menghasilkan citra biner yaitu citradengan hanya dua warna, yaitu: hitam dan putih. Operator ini memilih piksel yang memiliki nilai tertentu, atau lingkup tertentu. Proses ini dapat dilakukan apabila *brightness level* (atau *contrast*) telah diketahui dari gambar tersebut. Piksel yang *level-nya* lebih dari *threshold level*akan dirubah menjadi putih, dan sebaliknya piksel yang *level*-nya ada di bawah dari *levelthreshold* akan diubah menjadi hitam. Proses *thresholding* sering disebut dengan proses binerisasi. Pada beberapa aplikasi pengolahan citra, terlebih dahulu dilakukan threshold terhadap citra *gray level* untuk dapat menjadi citra biner (citra yang memiliki nilai level keabuan 0 atau 255) [9].

Gambar 4 Contoh *Tresholding*

Sumber : Dokumentasi pribadi, 25 Februari 2015

2.3.3 Dilasi

Dilasi adalah proses penumbuhan atau penebalan dalam citra biner yang merupakan proses penggabungan titik-titik latar (0) menjadi bagian dari objek (1), berdasarkan structuring element B yang digunakan. Jika A adalah citra input dan B adalah *structure element*, maka dilasi antara A dan B dinyatakan A $\oplus$ B dan didefinisikan dengan :

$$D(A,B) = A \oplus B \quad (2)$$

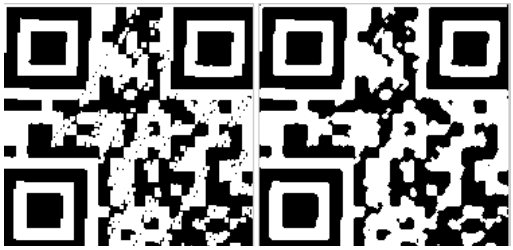
Keterangan :

$D(A,B)$ = Matriks hasil dilasi antara citra input (A) dengan*structuring element* (B)
A = Matriks citra *input*
B = *Structure element* 3x3 dengan nilai 1[10]

Gambar 5Perbandingan citra dilasi sebelum(kiri) dan sesudah(kanan)

2.3.3 *MedianFilter*

Medianfilter merupakan *filter* non-*linear* yang berfungsi untuk menghaluskan dan mengurangi *noise* atau gangguan pada citra. Operasi nonlinear dihitung dengan mengurutkan nilai intensitas sekelompok *pixel*, kemudian menggantikan nilai *pixel* yang diproses dengan nilai tertentu [11].



Gambar 6Citratanpa median filter (kiri) dengan median filter (kanan)

2.4 *Deblurring*

Teknik *deblurring* adalah sebuah prosedur yang dilakukan mengurangi *blur* dalam sebuah gambar kabur dan membuat sebuah citra yang telah tergradasikan menjadi tampak lebih baik sehingga menghasilkan sebuah citra yang lebih jernih. Dalam perancangan ini akan digunakan teknik *deblurring* menggunakan metode*EnchantedIterativeVanCittert*. Berikut adalah langkah-langkah dalam melakukan tahap*deblurring* :

2.4.1 *Point Spread Function*

Point Spread Function (*PSF*) disebut optical phenomenon, yaitu hasil keluaran dari tampilan titik kecil sistem pencitraan atau impuls dari cahaya. Dalam teknik

deblurring salah satu variabel yang perlu dikomputasikan terlebih dahulu adalah *PSF*. *PSF* secara mendasar adalah sebuah aproksimasi dari operator*distortion* (h) yang akan dikonvolusikan dengan citra original untuk menghasilkan citra *Blur*. Dalam perancangan ini *PSF* yang akan digunakan untuk melakukan teknik *deblurring* adalah *GaussianPSF* dengan ukuran *PSF* yaitu matriks 3x3 dan parameter *blur* (σ) *GaussianBlur* adalah 10. σ bernilai 10 adalah hasil percobaan menggunakan matlab dengan hasil *blur* yang signifikan mengalami perbedaan *blur* dengan iterasi sebelumnya. Rumus *GaussianPSF* adalah sebagai berikut :

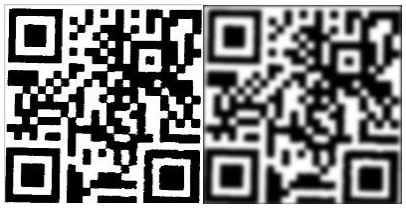
$$h_g(m,n) = e^{\frac{-(m^2+n^2)}{2 x \sigma^2}}$$

$$h\ (m,n) = \frac{h_g(m,n)}{\sum m \sum n\ h_g} \quad (3)$$

Dimana :

h_g	= Operator *Gaussian*
h	= *Gaussianblurfilter*
m	= Dimensi matriks m
n	= Dimensi matriks n
e	= Eksponen
σ	= Parameter *blur*

$\sum m \sum n\ \ h_g$ = Total seluruh pixel hg



Gambar 7Perbandingan Gambar Asli (kiri) Dengan Gaussian PSF (kanan)

2.4.2 *Enchanted Itterative Van Cittert*

Algoritma *Van* Cittert adalah sebuah algoritma Iterative yang terkenal didalam area *deblurring* pada citra. Algoritma ini mempunyai banyak keuntungan dalam kecepatan, mengandung sedikit variabel angka, operasi matematika yang sederhana, dan tidak memiliki batas penghalusan (*smoothness*). Berikut adalah rumus *Enchanted Iterative Van Cittert* yang dijelaskan sebagai berikut :

$$f^{(n+1)}=f^{n}+\beta(g-H\otimes f^{n}) \quad (4)$$

Dimana :

f^(n+1)	= Estimasi baru dari f^n
f^n	= Estimasi *Iterative*
n	= Jumlah langkah dalam *Iterative*
g	= Citra yang telah mengalami *blur*
H	= *GaussianPSF*
β	= Koefisien Sharpening
⊗	= Konvolusi

3. Pengujian Sistem

3.1. Persiapan Dokumen Uji Coba

Persiapan data dimulai dengan membangkitkan citra *QR-CODE* pada website https://www.the-qrcode-generator.com dan dicetak menggunakan printer HP-1000 kemudian diberikan *scratch*, *blur*, kombinasi *blur* dan *scratch*, dan kombinasi *scratch* dan *blur*. Total seluruh data percobaan adalah 300 buah.

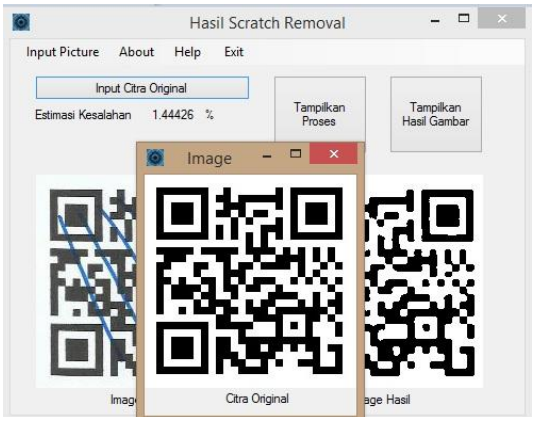
3.2. Tahapan Pembuatan

Langkah-langkah yang dilakukan pada tahap pembuatan sistem adalah:

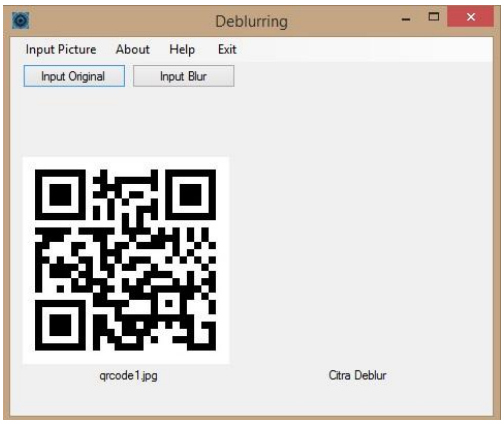
1. Melakukan identifikasi masalah
2. Menentukan tujuan perancangan
3. Melakukan analisis sistem
4. Melakukan perancangan sistem

Berikut akan ditampilkan *userinterfaceprogram* yang telah dibuat :

Gambar 8 Modul Main Menu



Gambar 9 Modul Scratch Removal



Gambar 10 Modul Deblurring

Gambar 11 Modul Combination

4. Hasil Percobaan

Pengujian terhadap sistem dilakukan dengan melakukan proses testing terhadap data *scratch*, *blur*, kombinasi *scratch* dan *blur*,kombinasi *blur* dan *scratch*. Untuk mengukur kualitas hasil program maka akan dilakukan perhitungan estimasi kesalahan yang akan dijelaskan pada rumus berikut :

$$E = \frac{\sum PO - \sum PA}{\sum PO} \text{ X } 100\% \quad (5)$$

Dimana :

E = Estimasi kesalahan

$\sum PO$ = Total seluruh nilai piksel citra *QR- CODE* tanpa kerusakan

$\sum PA$ = Total seluruh nilai piksel citra *QR-CODE* hasil keluaran program

Hasil pengujian terhadap data *scratch*akan ditampilkan pada Tabel 1

Tabel 1 Hasil Percobaan *Scratch Removal*

Jumlah Data	Skala Percobaan	Warna Spidol	Rata-rata Estimasi Kesalahan	Rata-rata keberhasilan pembacaan
10	Ringan	Biru	1.6259%	100%
10	Ringan	Ungu	1.3317%	100%
10	Ringan	Merah	1.4219%	100%
10	Sedang	Biru	1.3771%	100%
10	Sedang	Ungu	1.1017%	100%
10	Sedang	Merah	1.0078%	100%
10	Berat	Biru	1.6785%	100%
10	Berat	Ungu	1.1804%	100%
10	Berat	Merah	1.1983%	100%

Rata-rata estimasi kesalahan pada data percobaan scratch ringan dapat lebih besar dari data *scratch* sedang karena pada proses pemberian *scratch* tekanan spidol sangat berpengaruh terhadap hasil akhir program. Sehingga rata-rata estimasi kesalahan yang melibatkan proses *scratch* menghasilkan nilai yang sangat bergantung pada tingkat ketebalan tinta spidol. Karena pada proses *scratchremoval* menggunakan segmentasi warna (*HSV*) maka semakin gelap warna *scratch*akan semakin sulit untuk terdeteksi. Pada proses scan menggunakan printer HP-2060 citra *QR-CODE* yang telah diberikan scratch menjadi lebih terang (*contrast* bertambah) sehingga warna *scratch* menjadi lebih terang dan mempengaruhi estimasi kesalahan. Seluruh hasil percobaan (90 data) scratch removal mampu terbaca dengan baik menggunakan aplikasi QRCODE READER dengan persentase 100%.

Hasil pengujian terhadap data *blur*akan ditampilkan pada Tabel 2

Tabel 2Hasil Percobaan *Deblurring*

Jumlah Data	Skala Percobaan	Rata-rata Estimasi Kesalahan	Rata-rata keberhasilan pembacaan
10	Ringan	2.0918%	100%
10	Sedang	2.5198%	100%
10	Kuat	2.9535%	90%

Percobaan yang melibatkan proses *deblurring* pada persiapan data telah disertai dengan pra-penilitian yang telah dilampirkan pada. Seluruh hasil percobaan (30 data) blur mampu terbaca dengan baik menggunakan aplikasi QRCODE READER dengan persentase 90%.

Hasil pengujian terhadap data *Scratch* dan *blur*akan ditampilkan pada Tabel 3

Tabel 3 Hasil Percobaan *Scratch Removal* dan *Deblurring*

Jumlah Data	Skala Percobaan	Warna Spidol	Rata-rata Estimasi Kesalahan	Rata-rata keberhasilan pembacaan
10	Blur ringan scratch ringan	Biru	0.6207%	100%
10	Blur ringan scratch ringan	Ungu	2.5449%	100%
10	Blur ringan scratch ringan	Merah	1.0044%	100%
10	Blur ringan scratch sedang	Biru	2.3758%	100%
10	Blur ringan scratch sedang	Ungu	2.4792%	100%
10	Blur ringan scratch sedang	Merah	0.8038%	100%
10	Blur ringan scratch berat	Biru	2.462%	100%
10	Blur ringan scratch berat	Ungu	2.6025%	100%
10	Blur ringan scratch berat	Merah	0.7023%	100%

Hasil pengujian terhadap data *blur* dan *Scratch*akan ditampilkan pada Tabel 4

Tabel 4 Hasil Percobaan *Deblurring*dan *Scratch Removal*

Jumlah Data	Skala Percobaan	Warna Spidol	Rata-rata Estimasi Kesalahan	Rata-rata keberhasilan pembacaam
10	Scratch ringan blur ringan	Biru	5.0639%	100%
10	Scratch ringan blur ringan	Ungu	5.1317%	100%
10	Scratch ringan blur ringan	Merah	5.7409%	100%
10	Scratch sedang blur ringan	Biru	5.1798%	100%
10	Scratch sedang blur ringan	Ungu	6.264%	100%
10	Scratch sedang blur ringan	Merah	6.9111%	100%
10	Scratch berat blur ringan	Biru	4.9971%	100%
10	Scratch berat blur ringan	Ungu	6.3096%	100%
10	Scratch berat blur ringan	Merah	6.9909%	100%

Data percobaan *blurscratch* memiliki rata-rata hasil yang lebih kecil dibandingkan dengan data percobaan *scratchblur* karena pada saat proses scan, printer mampu menghasilkan sebuah citra yang sangat tajam dan mengurangi efek *blur*. Sedangkan pada data percobaan *scratch blur* estimasi kesalahan lebih besar karena efek *blur* menyebabkan warna-warna hitam pada *QR-CODE* menjadi melebar (dikaburkan dengan *GaussianPSF*) berwarna hitam kabur. Pada proses *scratchremoval* piksel-piksel yang berwarna hitam buram tersebut dipertegas menjadi warna hitam sehingga warna hitam pada *QR-CODE* mengalami penebalan. Seluruh hasil percobaan (180 data) kombinasi mampu terbaca dengan baik menggunakan aplikasi QRCODE READER dengan persentase 100%.

5. Kesimpulan

Hasil perancangan sistem *scratch removal* dan *deblurring* terhadap citra *QR-CODE* sudah berjalan dengan baik. Beberapa hal yang dapat disimpulkan dalam perancangan ini antara lain:

1. Sistem scratch removal sudah menghasilkan keluaran yang baik yaitu dengan rata-rata estimasi kesalahan sebesar 1.3354%
2. Sistem deblurring sudah menghasilkan keluaran yang baik yaitu dengan rata-rata estimasi kesalahan sebesar 2.5217%
3. Sistem kombinasi *deblurring* dan *scratch removal* sudah menghasilkan keluaran yang baik yaitu dengan rata-rata estimasi kesalahan sebesar 1.7560%
4. Sistem kombinasi *scratch removal* dan *deblurring* sudah menghasilkan keluaran yang baik yaitu dengan rata-rata estimasi kesalahan sebesar 5.8432 %
5. Pada persiapan data proses yang melibatkan printer mengurangi ketepatan program dalam menentukan estimasi kesalahan, karena pada proses *scan* dan *print* efek *blur* pada cita *QR-CODE*menjadi berkurang.

REFERENSI

[1] Sri Maharsi, Pengaruh Perkembangan Teknologi Informasi Terhadap Bidang Akuntansi Manajemen,…… http://citation.itb.ac.id/pdf/akun-petra/15673-15671-1-PB.pdf, 28 Januari 2015.

[2] Devi Indriasari et al., Analisis dan Perancangan Layanan Perpustakaan UAJY Berbasis Mobile dengan Memanfaatkan QR Code, http://e-journal.uajy.ac.id/5556/1/TF76301.pdf, 28 Januari 2015.

[3] Rumah Barcode, Pengertian Barcode,http://rumahbarcode.com/index.php/news/83-pengertian-barcode, 28 Januari 2015.

[4] Rinaldi Munir et al., Pengembangan Aplikasi QR Code Generator danQR Code Reader dari Data Berbentuk Image, http://informatika.stei.itb.ac.id/~rinaldi.munir/TA/Makalah_TA%20Pasca%20Nugraha.pdf, 28 Januari 2015.
[5] Robabdul, *QR Codes advantages and disadvantages*, http://www.robabdul.com/*QR-CODE*-advantages-and-disadvantages.asp, 29 Januari 2015
[6] Ida B. Putu, *Implementasi Qr-Codedan Algoritma Kriptografi Aes* , www.cs.unud.ac.id/files/file/skripsi/1384047523/F02_1384621183_0, 28 Januari 2015
[7] Peter Kieseberg, *QR Code Security*, https://www.sbaresearch.org/wp-content/uploads/publications/QR_Code_Security.pdf,24Februari 2015
[8] Antonius Salim, Pengaruh Warna Terhadap Classification, http://library.binus.ac.id/eColls/eThesisdoc/Bab2/2012-1-00523-mtif%202.pdf, 25 Februari 2015
[9] NonDgree, BAB II, http://digilib.its.ac.id/public/ITS-NonDegree-7548-7405040015-bab2.pdf, 2 Maret 2015
[10] Eko Prasetyo, Materi Kuliah Pengolahan Citra Digital, (Gresik: Fakultas Teknologi Informasi Universitas Muhamamdiyah 2011).
[11] Sony Syarifuddin, Analisis Filtering Citra Dengan Metode Mean Filter Dan Median, Filter,http://elib.unikom.ac.id/files/disk1/58/jbptunikompp-gdl-s1-2006-sonynuryad-2886-jurnal.doc, 2 Maret 2015