

# Analisis Metode SIFT dan SURF untuk Sistem Pendeteksi Gambar Termanipulasi Penyerangan *Copy-Move Forgery*

Regina Lionnie, Trie Maya Kadarina, Mudrik Alaydrus

*Teknik Elektro, Universitas Mercu Buana, Jakarta*

regina.lionnie.id@ieee.org, trie.maya@mercubuana.ac.id, mudrikalaydrus@mercubuana.ac.id

## **Abstrak**

Pemalsuan gambar digital telah menjadi salah satu taktik penyebar hoax yang merupakan ajang penyebar provokasi, menimbulkan kerusuhan dan kebencian. Untuk dapat memerangi pemalsuan gambar digital perlu diciptakan sebuah sistem yang dapat mendeteksi gambar tersebut merupakan gambar hasil manipulasi apa tidak. Pada penelitian ini gambar termanipulasi jenis penyerangan *copy-move forgery* dengan variasi penyekalaan dan rotasi telah dianalisa oleh metode SIFT dan SURF. Hasilnya kedua metode dapat mendeteksi gambar termanipulasi jenis penyerangan *copy-move forgery* dengan SIFT memberikan hasil dua kali lebih banyak kecocokan dibandingkan SURF dan SURF memberikan hasil pemrosesan waktu 0.33 kali lebih cepat dibandingkan SIFT.

**Keywords:** *copy-move forgery*, gambar termanipulasi, SIFT, SURF

## **1. PENDAHULUAN**

Pemalsuan gambar digital telah menjadi salah satu taktik penyebar hoax yang merupakan ajang penyebar provokasi, menimbulkan kerusuhan dan kebencian. Hoax sengaja disebar untuk memecah belah persatuan dan kesatuan terutama jika mengandung unsur SARA. Bentuk dari hoax ini dapat berupa kalimat yang diputar balikkan ataupun gambar yang sudah dimanipulasi atau didesain ulang [1]. Indonesia sebagai salah satu negara dengan tingkat konsumsi internet tertinggi di dunia [2] menjadi sarang empuk para penyebar hoax. Data dari Kemenkominfo menyebutkan bahwa terdapat sekitar 800.000 situs di Indonesia yang telah terindikasi sebagai penyebar informasi palsu [3]. Hal ini diperparah dengan teknologi dalam bidang editing gambar digital yang semakin mutakhir dan mudahnya mendapatkan perangkat lunak pengolah gambar digital.

Pemalsuan gambar digital dapat dikelompokkan menjadi dua jenis penyerangan yaitu penyerangan *copy-move forgery* dan *splicing* [4]. Pada penyerangan *copy-move*, manipulator akan melakukan penyalinan sebagian gambar dan meletakkannya di dalam gambar yang sama sehingga seolah-olah gambar yang diinginkan menjadi banyak atau ada bagian dari gambar yang diinginkan ditutupi. Gambar 1 memperlihatkan contoh gambar digital termanipulasi serangan tipe *copy-move*. Sedangkan pada penyerangan *splicing*, manipulator akan melakukan penyalinan sebagian gambar dan meletakkannya di gambar digital yang lain sehingga seolah-olah gambar tersebut menyajikan informasi yang tidak ada sebelumnya. Gambar 2 memperlihatkan contoh gambar digital termanipulasi serangan tipe *splicing*.



Gambar 1. Contoh Gambar Termanipulasi Serangan *Copy-Move*, bagian kiri gambar asli dan bagian kanan gambar yang terkena serangan [5].

Pada penelitian ini akan dilakukan studi perbandingan metode *scale-invariant feature transform* (SIFT) dan *speeded-up robust feature* (SURF) untuk melakukan pendeteksian pada gambar termanipulasi digital jenis penyerangan *copy-move forgery*. Simulasi akan dilakukan menggunakan perangkat lunak Matlab dengan komputer berkapasitas RAM 16Gb.

## 2. TINJAUAN PUSTAKA

### 2.1 Penelitian Terdahulu

Penelitian deteksi gambar termanipulasi jenis penyerangan *copy-move forgery* dimulai pada tahun 2017 menggunakan metode *scale-invariant feature transform* (SIFT) [6-7]. Pada [6], peneliti membandingkan beberapa jarak kedekatan kedua vektor SIFT pada perhitungan perkalian titik untuk membandingkan performa sistem pendeteksian. Sedangkan pada [7], peneliti membandingkan metode pra-proses seperti tapis lolos bawah dengan tapis median dan Gaussian, ekualisasi histogram dan tapis lolos atas, untuk membangun sistem pendeteksi gambar termanipulasi berdasarkan metode SIFT. Penelitian menggunakan metode analisis fitur lokal seperti SIFT dimulai di [8] dan disusul pengembangan metode SIFT seperti *Reflective SIFT* dan *MIFT (Mirror Reflection SIFT)* di [9-10]. Pada penelitian ini akan membandingkan dua metode yaitu SIFT dan SURF untuk gambar termanipulasi digital penyerangan *copy-move forgery*. Tabel 1 merupakan tabel perbandingan usulan penelitian ini dan perbandingan metode pada penelitian sebelumnya.

Tabel 1. Perbandingan Penelitian dan Penelitian Sebelumnya

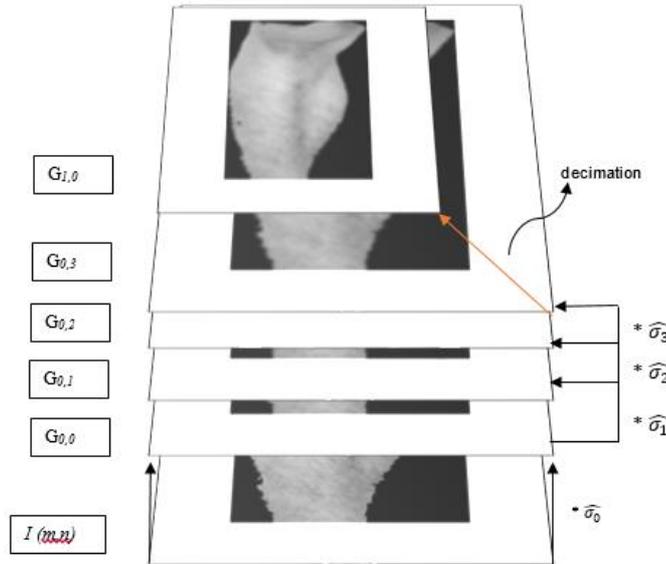
No	Penelitian	Metode Utama	Komparasi
1	Lionnie, et al [6]	SIFT	Jarak kedekatan vektor SIFT
2	Lionnie, et al [7]	SIFT	Metode pra proses (tapis lolos bawah, atas dan ekualisasi histogram)
3	Huang [8]	SIFT	-
4	Agarwal [9]	<i>Reflective SIFT</i>	-
5	Guo [10]	<i>MIFT (Mirror Reflective SIFT)</i>	-
6	<b>Penelitian ini</b>	<b><i>SIFT dan SURF</i></b>	<b>Waktu dan hasil kecocokan</b>

## 2.2 Scale-Invariant Feature Transform (SIFT)

Terdapat 4 langkah dalam algoritma SIFT [11,12] yang dijelaskan lebih detail dalam subbab berikut ini.

### 2.2.1 Mencari Ekstrema Skala-Ruang

Hal pertama yang perlu dilakukan adalah membangun ruang skala hierarki Gauss. Pada [13] telah dibangun ruang skala dengan proses kernel dan *sub-sampling* Gaussian. Ada 4 oktaf dan 3 tingkat / skala pada setiap oktaf seperti yang dapat dilihat pada Gambar 2.



Gambar 2. Skala Ruang Hierarki Gauss [13].

Skala ruang hierarki Gauss dapat diperoleh dengan konvolusi gambar asli  $I(m,n)$  with kernel Gauss  $H^{G,\hat{\sigma}_v}$ . Proses ini dapat dilihat pada persamaan (1) hingga (7) [12].

$$G(m,n,u=0,v=0) = (I * H^{G,\hat{\sigma}_0})(m,n) \tag{1}$$

$$G(m,n,u,v) = (G_{u,0} * H^{G,\hat{\sigma}_v})(m,n) \tag{2}$$

$$\hat{\sigma}_0 = \sqrt{\sigma_0^2 - \sigma_s^2} \tag{3}$$

$$\hat{\sigma}_v = \sigma_0 \cdot \sqrt{2^{2v/V} - 1} \tag{4}$$

dimana  $\sigma$  adalah lebar dari filter Gaussian dan diatur agar nilai  $\sigma_0 = 1.6$  dan  $\sigma_s = 0.5$  [11]. Pada persamaan (1) dan (2),  $(m,n)$  menunjukkan lokasi piksel (vertikal and horisontal) dalam sebuah gambar. Skala ruang hierarki Gauss dengan oktaf  $U$  yaitu

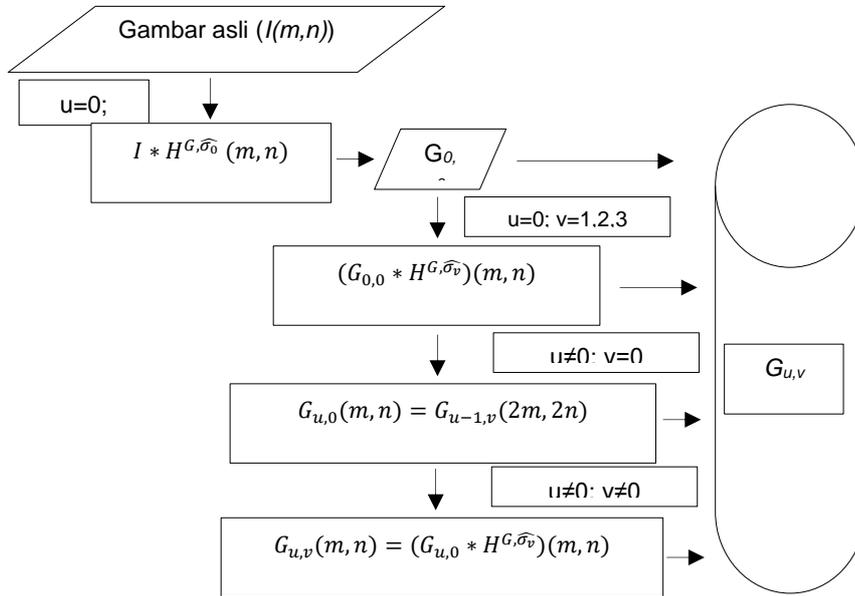
$$G = (G_0, G_1, \dots, G_{U-1}) \tag{5}$$

Dan setiap oktaf memiliki level  $V+1$  dengan skala  $G_{u,v}$

$$G_u = (G_{u,0}, G_{u,1}, \dots, G_{u,v}) \tag{6}$$

dimana  $u \in [0, U - 1]$  menunjukkan indeks oktaf dan  $v \in [0, V]$  menunjukkan indeks level pada oktaf yang sama. Dalam penelitian ini, akan diset nilai  $U=4$  dan  $V=3$ .

Diagram alir dari proses skala ruang hierarki Gauss diilustrasikan pada Gambar 3. Proses ini dimulai dari melakukan pra filter dari matriks gambar asli dengan kernel Gauss dengan lebar  $\sigma_s = 0.5$  and set nilai skala basis untuk setiap oktaf dengan  $\sigma_0 = 1.6$  [11]. Proses ini akan membangun basis dari oktaf  $(G_{0,0})$ .



Gambar 3. Diagram Alir Skala Ruang Hierarki Gauss dengan oktaf  $U$  dan level  $V$  [13].

Level berikutnya pada oktaf yang sama (oktaf basis) didapatkan dari konvolusi level dasar dengan Gaussian kernel dengan lebar  $\hat{\sigma}_v$ . Pada persamaan (4),  $\hat{\sigma}_v$  dapat digunakan di setiap oktaf karena karakteristiknya yang independen dari oktaf indeks  $u$ . Saat nilai  $v$  mencapai nilai terakhir, yaitu 3 ( $G_{u,3}$ ), proses berlanjut ke oktaf selanjutnya ( $G_{u+1,0}$ ) dengan melakukan proses desimasi dari oktaf terakhir dan level terakhir dan *sub-sampling* dengan faktor 2 yang dirangkum pada persamaan (7).

$$G_{u,0}(m, n) = G_{u-1,q}(2m, 2n) \tag{7}$$

Pada level 0 di setiap oktaf tidak dilakukan filter dengan Gauss kembali, dan hanya dilakukan proses desimasi dari tingkat level terakhir di oktaf sebelumnya. Dilakukan filter kembali pada level 1 di setiap oktaf. Proses ini terus berlanjut hingga tercapai oktaf dan level yang diinginkan. Pada Gambar 4, merupakan ilustrasi pembuatan skala ruang hingga  $G_{1,0}$ . Saat level pada setiap oktaf mencapai level tertinggi, proses akan dilanjutkan dengan proses desimasi (*sub-sampling* 2:1) dan hasil proses desimasi ini adalah level dasar ( $v=0$ ) untuk oktaf selanjutnya. Dari level tertinggi dari oktaf sebelumnya dan level dasar dari oktaf selanjutnya tidak dilakukan proses konvolusi dengan filter Gauss dan proses filter konvolusi ini akan dimulai lagi dari level dasar ( $G_{u,0}$ ) menuju level 1 ( $G_{u,1}$ ) di setiap oktaf.

Setelah berhasil membangun ruang skala hierarki Gauss, dibentuk *laplacian of Gaussian* dengan menggunakan kernel Gauss seperti pada persamaan (8).

$$H(m, n, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{(m^2+n^2)}{2\sigma^2}} \tag{8}$$

Untuk membentuk *laplacian of Gaussian* (LoG) dibutuhkan usaha komputasi yang besar dan tidak efisien. Oleh karena itu, biasanya dilakukan pendekatan dengan aproksimasi LoG (*laplacian of Gaussian*) dengan menggunakan DoG

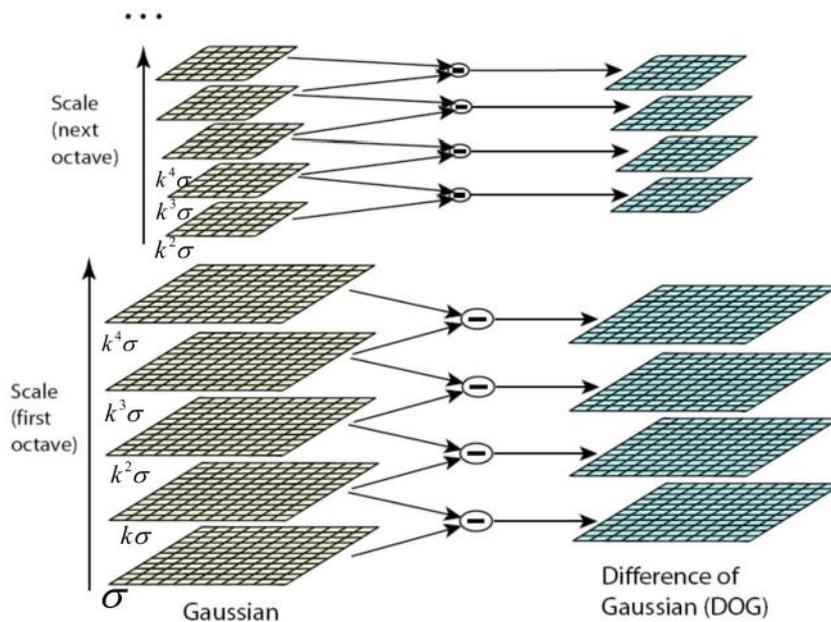
(*difference of Gaussian*). Dengan menggunakan DoG lebih menghemat dalam hal efisiensi komputasi dan menyediakan aproksimasi yang baik untuk LoG. Persamaan mendapatkan DoG dapat ditemukan di (9). Persamaan (10) hingga (12) menunjukkan proses aproksimasi dari LoG ke DoG. Gambar 4 menunjukkan cara mendapatkan (DoG) dari skala ruang.

$$D(m, n, \sigma) = [H(m, n, k\sigma) - H(m, n, \sigma)] * I(m, n) = G(m, n, k\sigma) - G(m, n, \sigma) \quad (9)$$

$$\frac{\partial H}{\partial \sigma} = \sigma \Delta^2 H \quad (10)$$

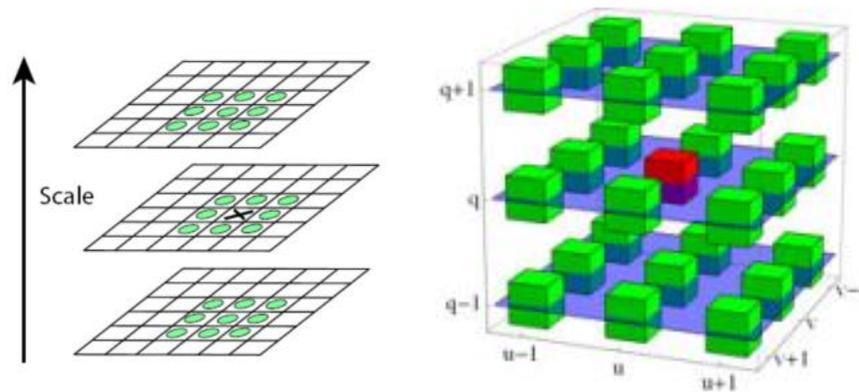
$$\sigma \Delta^2 H = \frac{\partial H}{\partial \sigma} = \frac{H(m, n, k\sigma) - H(m, n, \sigma)}{k\sigma - \sigma} \quad (11)$$

$$H(m, n, k\sigma) - H(m, n, \sigma) \approx (k - 1)\sigma^2 \Delta^2 H \quad (12)$$



Gambar 4. Proses Mendapatkan DoG dari Skala Ruang [11]

Dari Gambar 4 dapat dilihat setiap DoG didapatkan dengan mengurangi skala yang dituju dengan skala sebelumnya (pengurangan  $k^n\sigma$  dengan  $k^{n-1}\sigma$ ). Setiap DoG mempunyai ukuran yang sama dengan skala ruang dari oktaf yang sama. Setelah berhasil membentuk DoG untuk skala ruang hierarki Gauss, harus didapatkan nilai ekstrema (maksimum ataupun minimum) dengan cara membandingkan sebuah nilai piksel (tengah) dengan 26 nilai piksel dengan rincian 8 piksel di level yang sama dan 18 piksel di level sebelum dan sesudah level tujuan. Ilustrasi proses ini dapat dilihat pada Gambar 5.



Gambar 5. Mendeteksi Lokal Ekstrema [7].

Dalam Gambar 5,  $x$  (kiri) atau piksel bertanda merah (kanan) menunjukkan piksel acuan yang akan dibandingkan dengan 9 piksel dari level sebelumnya dan 9 piksel dari level sesudahnya bersama dengan 8 piksel pada level yang sama. Terdapat 26 piksel total yang akan dibandingkan dengan piksel acuan. Jika nilai piksel acuan lebih besar ataupun lebih kecil dibandingkan 26 nilai piksel ini maka piksel acuan merupakan ekstrema dan selanjutnya akan disebut sebagai kandidat keypoint.

### 2.2.2 Lokalisasi Keypoint dan Filtering

Setelah mendapatkan kandidat keypoint, kandidat ini akan disaring kembali menggunakan metode ekspansi Taylor dan matriks Hessian. Pengurangan jumlah keypoint ini dipilih dari kandidat yang memiliki kontras yang buruk dan merupakan pilihan kandidat yang terdapat disepanjang tepian. Yang dilakukan adalah melakukan pencarian lokasi, skala dan rasio dari *principal curvature* dengan menggunakan ekspansi Taylor. Dari persamaan (9) mendapatkan DoG, akan diaplikasikan ekspansi Taylor yang dapat dilihat pada persamaan (13), dimana  $D$  dan turunannya dievaluasi pada kandidat keypoint dan  $x = (x, y, \sigma)^T$  adalah *offset* kandidat keypoint ini.

$$D(x) = D + \frac{\partial D^T}{\partial x} x + \frac{1}{2} x^T \frac{\partial^2 D}{\partial x^2} x \quad (13)$$

Lokasi dari ekstrema,  $\hat{x}$ , ditentukan dengan mencari turunan dari (13) dan memberikan nilainya dengan nol sehingga menghasilkan persamaan (14).

$$\hat{x} = -\frac{\partial^2 D^{-1} \partial D}{\partial x^2 \partial x} \quad (14)$$

Ganti (14) ke dalam persamaan (13) memberikan (15) dan pada [11] jika  $|D(\hat{x})| < 0.03$ , ekstrema akan dihilangkan dan mengurangi total keypoint.

$$D(\hat{x}) = D + \frac{1}{2} \frac{\partial D^T}{\partial x} \hat{x} \quad (15)$$

Dikarenakan tidak cukup dengan menolak kandidat keypoint yang memiliki kontras yang tidak baik, akan dihitung juga *principal curvature* dari DoG. DoG akan memiliki respon yang kuat sepanjang tepian dan beberapa nilai derau yang tidak stabil. Sepanjang tepian DOG akan memiliki nilai *principal curvature* yang besar dan dengan arah tegak lurus akan memiliki nilai yang kecil. Dalam penelitian ini akan dihitung *principal curvature* matriks Hessian.

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix} \quad (16)$$

Dengan  $D$  menunjukkan turunan dengan menghitung perbedaan nilai keypoint dengan tetangganya. Kemudian, dihitung nilai eigen dari  $H$ . Jika  $\alpha$  adalah nilai eigen terbesar dari  $H$  dan  $\beta$  nilai eigen terkecil dari  $H$ , dapat dihitung jumlah nilai eigen dari  $trace$  (17) dan hasil perkaliannya dari determinan (18).

$$Tr(H) = D_{xx} + D_{yy} = \alpha + \beta \tag{17}$$

$$Det(H) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta \tag{18}$$

Jika  $r = \frac{\alpha}{\beta}$  maka

$$\frac{Tr(H)^2}{Det(H)} = \frac{(\alpha+\beta)(\alpha+\beta)}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta\beta} = \frac{r^2\beta^2+2r\beta+\beta^2}{r\beta^2}$$

$$\frac{Tr(H)^2}{Det(H)} = \frac{(r+1)^2}{r} \tag{19}$$

Jika  $\alpha = \beta$  maka  $r = 1$ , persamaan (19) adalah minimum dan nilainya akan bertambah sepanjang nilai  $r$ . Untuk mengurangi keypoint lebih lanjut, pada [11] penulis menentukan nilai ambang: jika pada persamaan (19) nilai rasio lebih besar dari 10, maka kandidat keypoint dihilangkan.

### 2.2.3 Menentukan Orientasi

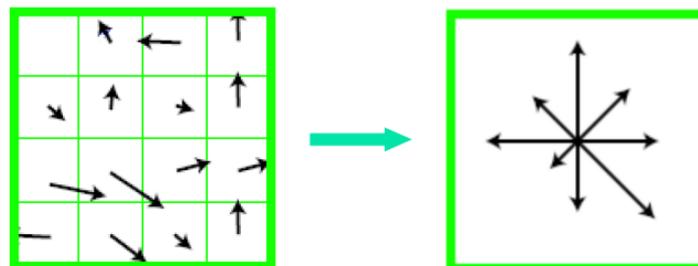
Langkah ketiga adalah mencari orientasi dari keypoint. Tujuan dari langkah ini adalah mencari orientasi dominan dan membuat lurus (*align*) point lainnya dalam area yang sama dengan orientasi ini. Hal ini merupakan hal yang unik karena merepresentasikan area disekitar keypoint. Hal ini juga dilakukan karena akan dicapai *rotation invariance* (tidak berubah jika terjadi rotasi). Untuk mendapatkan orientasi disekitar keypoint, pertama harus dihitung magnituda  $m(x,y)$  (20) dan arah  $\theta(x,y)$  (21).

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \tag{20}$$

$$\theta(x,y) = \tan^{-1}\left(\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}\right) \tag{21}$$

Setelah mengetahui magnituda dan arahnya, akan dihitung histogram orientasi dengan pemberat dalam area keypoint (dengan pemberat  $\sigma = 1.5$  dan total 36 bin) [11]. Jika digunakan 36 bin, maka 1 bin akan mencakup  $360^\circ/36 = 10^\circ/\text{bin}$ . Gambar 6 (kiri) menunjukkan orientasi di area keypoint dengan magnituda dan arahnya dan Gambar 6 (kanan) menunjukkan pengutusan magnituda dan arahnya ke dalam histogram dengan 8 bin.

Hingga pada langkah ini, terdapat 36 bin histogram dengan masing-masing magnituda dan arahnya, akan dipilih satu dengan puncak tertinggi dan puncak tertinggi menunjukkan orientasi dominan pada keypoint.

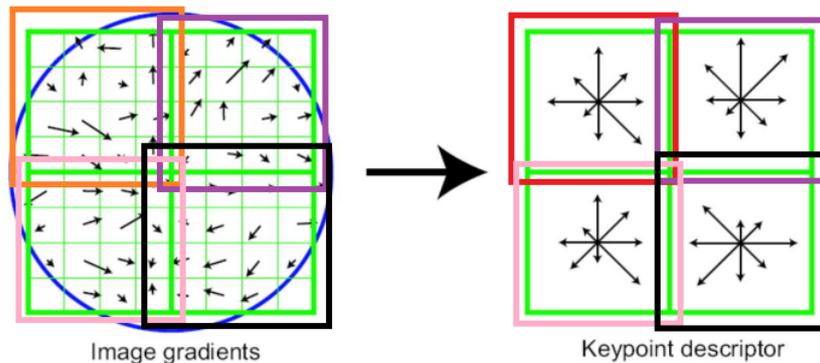


Gambar 6. Proses Orientasi Keypoint: magnituda dan arah untuk kernel of 4x4 (kiri) dan orientasi 8 bin (kanan) [6].

### 2.2.4 Membentuk SIFT deskriptor

Langkah terakhir algoritma SIFT adalah membuat deskriptor di sekitar keypoint. Terdapat kemungkinan untuk menggunakan dan menyimpan nilai intensitas keabuan dan nilai warna di sekitar keypoint akan tetapi ini adalah deskriptor dengan sifat yang lemah karena jika terdapat perubahan pencahayaan dan variasi iluminasi. Akan lebih baik jika menggunakan histogram orientasi gradien untuk menggambarkan area disekitar keypoint karena ini adalah representasi yang kuat. Gradien tidak akan banyak berubah dibandingkan dengan nilai intensitas keabuan yang akan berubah banyak jika ada variasi iluminasi.

Untuk membentuk deskriptor SIFT, dalam area keypoint dengan kernel  $16 \times 16$  akan dibagi dalam histogram dengan 8 bin dan setiap bin memiliki area  $4 \times 4$ . Gambar 7 menunjukkan contoh area  $8 \times 8$  yang dibagi dalam area  $2 \times 2$ . Jika satu area  $4 \times 4$  memiliki gradien orientasi histogram dengan 8 bin maka  $16 \times 16$  area akan memiliki 16 histogram orientasi dengan masing-masing 8 bin setiap histogram. Hingga langkah ini telah terdapat  $16(\text{histogram}) \times 8(\text{bin}) = 128$  deskriptor yang selanjutnya akan disebut fitur deskriptor yang akan disusun menjadi vektor kolom untuk tujuan langkah selanjutnya yaitu klasifikasi.



Gambar 7. Pembentukan deskriptor keypoint area  $8 \times 8$  menjadi area  $2 \times 2$  gradien orientasi histogram.

### 2.3 Speeded-up Robust Feature Transform (SURF)

Dalam manipulasi gambar digital termanipulasi jenis penyerangan *copy move forgery*, tugas algoritma dalam mencari koresponden diantara area dalam gambar yang terkena serangan menjadi obyek penelitian dalam teknik ilmu *computer vision*. Proses pencarian korespondensi ini terbagi menjadi tiga langkah utama. Langkah pertama yaitu proses seleksi titik kunci (*interest point detection*) pada lokasi yang berbeda seperti di sudut, dan di persimpangan obyek di dalam gambar. Karakter terpenting pada titik kunci adalah kestabilan titik kunci di berbagai skala gambar yang sama. Langkah kedua adalah bagaimana cara mendeskripsi tetangga dari titik kunci ini (*interest point description*) yang direpresentasikan dalam fitur vektor. Langkah terakhir adalah proses *matching* diantara fitur vektor berdasarkan jaraknya. Metode SIFT [11] yang dikembangkan David Lowe mencakup kedua proses yaitu *interest point detection* dengan diferensiasi Gaussian yang melakukan aproksimasi untuk *Laplacian Gaussian* dan *interest point description* dengan histogram dari lokal gradien. Metode SIFT dapat mendeteksi fitur lokal yang stabil dalam tingkat skala gambar yang berbeda-beda, tahan terhadap transformasi geometri dan variasi pencahayaan. Akan tetapi, metode ini cukup memakan waktu

yang lama untuk dapat bekerja dengan baik. H. Bay et al [14] mengembangkan metode SURF berdasarkan proses *interest point detection* dan *interest point descriptor* yang dapat menyamai bahkan melebihi performa SIFT akan tetapi dalam waktu komputasi yang lebih singkat.

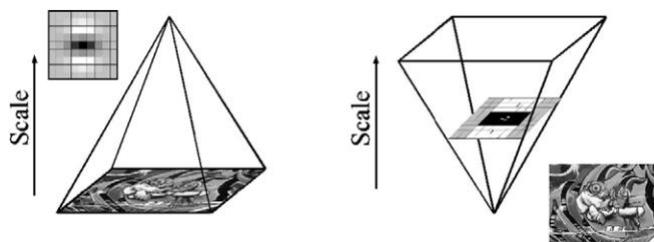
Metode SURF menggunakan pendekatan *integral images* dan Hessian matriks untuk mencari titik kunci yang stabil dalam berbagai tingkat penyekalaan. Persamaan (22) merepresentasikan perhitungan *integral images* dimana total dari penjumlahan piksel suatu area di gambar ( $I$ ) dapat diproses secara cepat dengan algoritma ini.

$$I_{\Sigma(x)} = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq y} I(i, j) \quad (22)$$

Hessian matrix pada SURF digunakan karena akurasi dari perhitungan matriksnya dalam mendeteksi struktur seperti *blob* di lokasi dimana perhitungan determinan dari matriksnya merupakan nilai maksimum. Persamaan (23) menunjukkan perhitungan matriks Hessian ( $H(x)$ ) pada sebuah gambar digital ( $I$ ) dengan  $L_{xx}$  adalah konvolusi turunan kedua Gaussian dengan gambar asli dengan lebar pita  $\sigma=1,2$  [14].

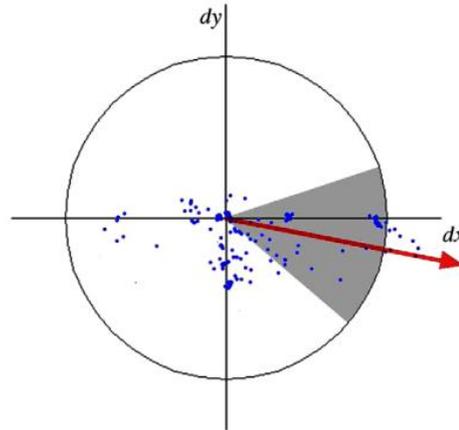
$$H(x, \sigma) = \begin{bmatrix} L_{xx}(x, \sigma) & L_{xy}(x, \sigma) \\ L_{xy}(x, \sigma) & L_{yy}(x, \sigma) \end{bmatrix} \quad (23)$$

Agar titik kunci stabil di setiap level penyekalaan, akan dilakukan analisis dimana ketika level penyekalaan meningkat, ukuran tapis akan bertambah, tidak seperti reduksi ukuran gambar seperti yang dikerjakan di [11]. Gambar 8 di bawah memperlihatkan proses pelebaran filter ketika skala bertambah.



Gambar 8 Proses Pelebaran Ukuran Filter Ketika Level Penyekalaan Bertambah pada SURF (kanan) dan Reduksi Ukuran Gambar Digital pada SIFT (kiri) [14]

Untuk proses *interest point descriptor*, metode SURF menggunakan algoritma transformasi wavelet Haar tingkat 1 dalam arah vertikal dan horisontal. Setelah mendapatkan respon dari transformasi ini, akan dilakukan *sliding orientation* sebesar 60 derajat untuk mencari arah orientasi yang paling dominan seperti terlihat pada Gambar 9 berikut. Titik kunci, lokasi dan orientasi dominan ini akan dimasukkan ke dalam fitur vektor dengan total 64 dimensi yang akan dicari kedekatannya dengan perhitungan hasil kali titik.

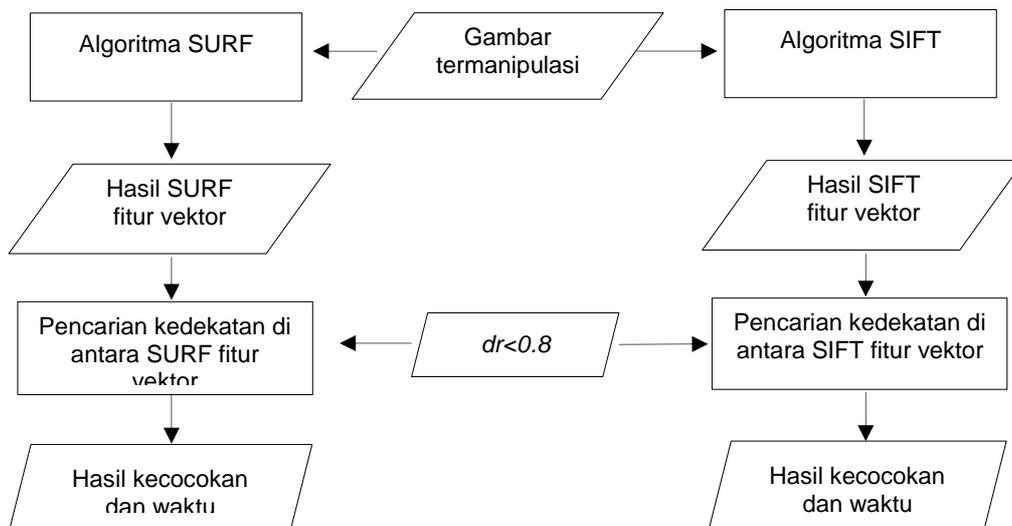


Gambar 9 *Sliding Orientation* setiap 60 Derajat [14].

Proses *matching* dari fitur vektor SURF akan dilakukan menggunakan proses hasil kali titik cosinus yang menghasilkan keluaran seperti metode tetangga terdekat tetapi dengan proses waktu komputasi yang lebih cepat. Hasil perbandingan kedekatan diantara tetangga terdekat pertama dan kedua akan memperlihatkan bahwa gambar uji sudah termanipulasi atau tidak, dengan aturan jarak kedekatan kurang dari 0.8 [11].

### 3. DESAIN PENELITIAN

Desain penelitian dapat dilihat pada Gambar 10 di bawah ini. Penelitian ini akan membandingkan kedua metode SIFT dan SURF untuk jarak kedekatan ( $dr$ ) dibawah 0.8. Jarak kedekatan adalah jarak perbandingan dimana hasil kedekatan pertama dan kedekatan kedua dari perhitungan tetangga terdekat. Hasil dari algoritma pada penelitian ini yang akan dibandingkan adalah kecocokan dan waktu proses. Terdapat tiga jenis gambar yang sudah dimanipulasi. Gambar 11 [6] memperlihatkan gambar original (11.a) dan ketiga gambar termanipulasi, gambar manipulasi tanpa perubahan ukuran dan arah (11.b), gambar manipulasi dengan memperkecil dan perubahan arah 90 derajat (11.c) dan gambar manipulasi dengan memperbesar dan perubahan arah 180 derajat (11.d).



Gambar 10 Diagram Alir Desain Penelitian



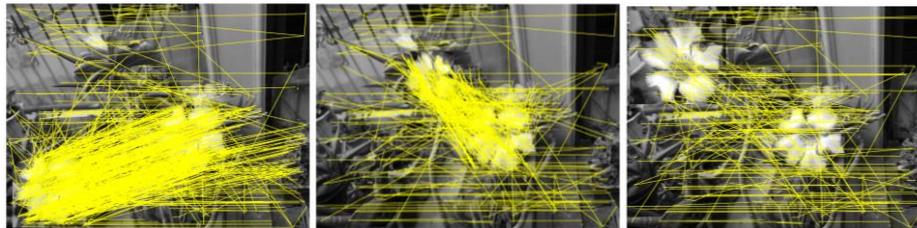
Gambar 11 (a) Gambar Asli dan (b,c,d) Gambar Termanipulasi [6]

#### 4. SIMULASI

Simulasi pada penelitian ini menggunakan Matlab 2016a dengan komputer dengan memori RAM 16GB. Tabel 2 memperlihatkan hasil simulasi pada penelitian ini. Gambar 12 memperlihatkan hasil penelitian dengan SIFT dan Gambar 13 untuk metode SURF.

Tabel 2. Hasil Penelitian dengan SIFT dan SURF

	Gambar 10.b	Gambar 10.c	Gambar 10.d
Hasil <i>Match</i> SIFT	930	584	400
Waktu Proses SIFT	35,572 detik	33,915 detik	37,428 detik
Hasil <i>Match</i> SURF	409	271	256
Waktu Proses SURF	1,123 detik	1,086 detik	1,079 detik



Gambar 12 (kiri) hasil SIFT untuk gambar 10.b (tengah) hasil SIFT untuk gambar 10.c dan (kanan) hasil SIFT untuk gambar 10.d.



Gambar 13 (kiri) hasil SURF untuk gambar 10.b (tengah) hasil SURF untuk gambar 10.c dan (kanan) hasil SURF untuk gambar 10.d.

Hasil penelitian memperlihatkan bahwa, kedua metode, SIFT dan SURF dapat digunakan untuk mendeteksi gambar termanipulasi dengan variasi manipulasi skala dan rotasi. SIFT memberikan hasil kecocokan lebih banyak sekitar 2 kali lebih banyak dibandingkan dengan SURF tetapi SURF dapat memproses sistem pendeteksi hampir 0.33 kali lebih cepat dibandingkan dengan SIFT untuk gambar yang sama.

## 5. KESIMPULAN

Sistem Pendeteksi Gambar termanipulasi digital dengan variasi rotasi dan penyekalaan telah dibangun menggunakan metode SIFT dan SURF. Hasil penelitian menunjukkan bahwa SIFT memberikan hasil kecocokan lebih banyak hampir dua kali lebih banyak dibandingkan dengan SURF akan tetapi SURF dapat memproses algoritma lebih cepat hingga hampir 0.33 kali lebih cepat dibandingkan dengan metode SIFT.

## REFERENCES

- [1] M. Nur, "Meresahkan, Polisi Soroti Bisnis Hoax," <https://www.jawapos.com/read/2018/04/25/207237/meresahkan-polri-soroti-bisnis-hoax>, Diakses 9 Mei 2018.
- [2] Internet World Stats, <https://www.internetworldstats.com/top20.htm>
- [3] A. Yuliani, Ada 800.000 Situs Penyebar Hoax di Indonesia, [https://kominfo.go.id/content/detail/12008/ada-800000-situs-penyebar-hoax-di-indonesia/0/sorotan\\_media](https://kominfo.go.id/content/detail/12008/ada-800000-situs-penyebar-hoax-di-indonesia/0/sorotan_media), Diakses 21 Juni 2018.
- [4] O. M. Al-Qershi dan Bee Ee Khoo, Passive detection of copy-move forgery in digital images: State-of-the-art, *Forensic science international*, 231.1 (2013): 284-295.
- [5] I. Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, G. Serra. A SIFT-based Forensic Method for Copy-Move Attack Detection and Transformation Recovery, *IEEE Transactions on Information Forensics and Security*, 2011, vol. 6, issue 3, pp. 1099-1110.
- [6] R. Lionnie dan M. Alaydrus., Sistem Pendeteksi Gambar Termanipulasi Menggunakan Metode SIFT, *Techné: Jurnal Ilmiah Elektroteknika* 16.02 (2017): 133-140.

- [7] R. Lionnie, et al., A study on pre-processing methods for copy-move forgery detection based on SIFT, In *Region 10 Conference, TENCON 2017*, pp. 1142-1147.
- [8] H. Huang, W, Guo, and Y. Zhang, Detection of Copy-Move Forgery in Digital Images Using SIFT Algorithm, *Pacific-Asia Workshop on Computational Intelligence and Industrial Application*, 2008, vol. 2, pp. 272-276.
- [9] V. Agarwal, and V. Mane, Reflective SIFT for Improving the Detection of Copy-Move Image Forgery, *Proc. Of Second International Conference on Research in Computational Intelligence and Communication Networks*, 2016.
- [10] X. Guo, et al., MIFT: A Mirror Reflection Invariant Feature Descriptor, *Computer Vision–ACCV*, 2010, pp. 536-545.
- [11] D. G. Lowe, Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*. 2004; 60(2): 91-110.
- [12] W. Burger, Burge MJ. *Principles of Digital Image Processing: Advanced Methods*. London: Springer-Verlag. 2013: 229-296.
- [13] R. Lionnie and M. Alaydrus, “Hierarchical Gaussian Scale-Space on Androgenic Hair Pattern Recognition,” *TELKOMNIKA* Vol. 15, No. 1, March 2017.
- [14] H. Bay et al, “Speeded-Up Robust Features (SURF),” *Computer Vision and Image Understanding 110* (2008), pp. 346-39.