FEATURE RECOGNITION BERBASIS *CORNER DETECTION* DENGAN METODE *FAST, SURF,* DAN *FLANN TREE* UNTUK IDENTIFIKASI LOGO PADA *AUGMENTED REALITY MOBILE SYSTEM*

Rastri Prathivi[1], Vincent Suhartono[2], Guruh Fajar Shidik[3]
[123]Pascasarjana Teknik Informatika Universitas Dian Nuswantoro

ABSTRACT

Logo is a graphical symbol that is the identity of an organization, institution, or company. Logo is generally used to introduce to the public the existence of an organization, institution, or company. Through the existence of an agency logo can be seen by the public. Feature recognition is one of the processes that exist within an augmented reality system. One of uses augmented reality is able to recognize the identity of the logo through a camera. The first step to make a process of feature recognition is through the corner detection. Incorporation of several method such as FAST, SURF, and FLANN TREE for the feature detection process based corner detection feature matching up process, will have the better ability to detect the presence of a logo. Additionally when running the feature extraction process there are several issues that arise as scale invariant feature and rotation invariant feature. In this study the research object in the form of logo to the priority to make the process of feature recognition. FAST, SURF, and FLANN TREE method will detection logo with scale invariant feature and rotation invariant feature conditions. Obtained from this study will demonstration the accuracy from FAST, SURF, and FLANN TREE methods to solve the scale invariant and rotation invariant feature problems.

Keywords: Logo Identification; Feature Recognition

1. PENDAHULUAN

1.1. Latar Belakang

Logo adalah simbol berbentuk grafis yang merupakan identitas dari sebuah organisasi, lembaga atau perusahaan. Logo umumnya digunakan untuk memperkenalkan kepada *public* keberadaan sebuah organisasi, lembaga atau perusahaan. Logo juga menjadi simbol jaminan mutu bagi sebuah hasil produksi dan telah menjadi keharusan yang wajib dimiliki oleh sebuah organisasi, lembaga atau perusahaan. Keberadaan logo di dalam sebuah organisasi, lembaga atau perusahaan menjadi penting karena melalui logo maka ciri khas atau informasi yang berkaitan dengan organisasi, lembaga atau perusahaan tertentu dapat diketahui. Informasi yang terkandung di dalam sebuah logo dapat disampaikan menggunakan media komunikasi seperti *mobile device* dan internet menggunakan aplikasi *augmented reality.* Untuk bisa mengenali bentuk sebuah logo di dalam sistem *augmented reality* maka terlebih dahulu perlu dilakukan proses *recognition* terhadap logo tersebut.

Berkaitan dengan jenis *image* yang dapat dideteksi di dalam proses *recognition,* maka ada dua jenis *image* yaitu *bitmap* dan *vector*. Logo termasuk ke dalam kategori *vector image*. Tingkat resolusi di dalam sebuah *image* berbentuk logo tidak memiliki peranan yang penting. Proses *recognition* yang diterapkan pada jenis *image* yang berbeda akan mempengaruhi keakuratan proses *recognition*. Proses *recognition* merupakan proses awal yang perlu ada untuk membangun sistem *augmented reality*. Proses *recognition* ini bertujuan untuk mengidentifikasi bentuk logo agar dapat dikenali, dideteksi dan dideskripsikan oleh sistem. Di dalam *recognition* yang berbasis *feature*, proses awal yang perlu dilakukan adalah melakukan

pendeteksian *corner*. Pendeteksian *corner* akan mendeteksi bentuk sebuah logo yang berorientasi pada sudut dalam logo tersebut. Logo akan sulit dideteksi jika letak sudut saat dideteksi tidak sesuai dengan data yang ada.

Selain itu letak sudut akan berbeda jika ada perbedaan kondisi tampilan seperti perbedaan atau perubahan skala dan rotasi. Maka perlu adanya metode *corner recognition* yang tepat untuk mengatasi masalah tersebut.

Proses *recognition* berbasis *corner feature* perlu mengalami tiga tahapan yaitu *feature detection* dan *extraction, feature descriptor* dan *feature matching*. Setiap tahapan tersebut dapat memiliki metode tertentu yang dapat dijadikan acuan.

Dalam *state of the art* tentang *recognition* berbasis *corner feature* untuk sistem *augmented reality* ada beberapa penelitian yang telah membuktikan penggunaan beberapa metode yang berbeda-beda antara lain Mikolajczyk, K. dan Schmid, C. [1]. Di dalam *papernya* yang berjudul *Performance Evaluation of Local Descriptor* dia melakukan penelitian terhadap beberapa *feature descriptor* dan menemukan bahwa *GLOH* (*Gradient Location and Orientation Histogram*) *descriptor* yang merupakan variant dari *SIFT* (*Scale Invariant Feature Transformation*) *descriptor* memiliki komputasi yang tinggi meskipun dapat mengatasi masalah pada perbedaan skala lebih baik daripada *SIFT descriptor*. Kelemahan dari penelitian ini adalah belum dapat mengatasi masalah pada kondisi perbedaan rotasi di dalam sebuah *image* dan hanya melakukan penelitian terhadap *feature descriptor*.

Wei-Chao Chen, Yingen Xiong, Jiang Gao, Natasha Gelfand, dan Radek Grzeszczuk [2], di dalam *papernya* yang berjudul *Efficient Extraction of Robust Image Features on Mobile Devices*, melakukan penelitian terhadap SURF untuk tahapan *feature extraction* dan *feature descriptor* serta *ANN* (*Aproximate Nearest Neighbor*) untuk tahapan *feature matching*. Penelitian ini menghasilkan komputasi rata – rata sebesar 30% lebih cepat dibandingkan dengan penelitian yang dilakukan oleh Herbert Bay, Tinne Tuytelaars, Luc Van Gool [3]. Objek penelitian mereka adalah *image* berjenis bitmap atau foto.

Gabriel Takacs [4] di dalam papernya yang berjudul *Outdoors Augmented Reality on Mobile Device Using Loxel-Based Visual Feature Organization* membangun sistem *outdoor augmented reality* dengan memanfaatkan proses *image recognition* di sisi *client* dan proses *matching* di sisi *server*. Mereka menggunakan metode *SURF*, ANN, dan Brute Force untuk melakukan proses *recognition* di sisi *client* pada *mobile device*. Objek penelitian mereka adalah *real time video* yang diambil secara langsung.

Niels Henze, Torben Schinke dan Susanne Boll [5] di dalam papernya yang berjudul *What is That? Object Recognition from Natural Features on a Mobile Device* menggunakan penggabungan metode yang berbeda – beda untuk setiap tahapan proses pada *recognition*. Mereka menggunakan metode *FAST*, *SIFT* dan Vocabulary Tree. Objek penelitian mereka adalah poster yang berukuran 45x55 cm. Proses *recognition* yang diterapkan berbasis *natural features* yaitu melakukan pendeteksian *image* sesuai dengan bentuk *image*nya.

Daniel Wagner [6] dan teman-teman di dalam papernya yang berjudul *Real Time Detection and Tracking for Augmented reality on Mobile Device* menyatakan bahwa metode *SIFT* sangat kuat untuk melakukan proses *recognition* tetapi memerlukan komputasi yang tinggi. Mereka melakukan modifikasi pada metode *SIFT* sehingga dapat diimplementasikan pada mobile *device*. Mereka menggunakan metode *FAST* pada proses *interest point detection* dan ekstraksi *image*, proses *feature descriptor* menggunakan metode *SIFT* dan proses *feature matching* menggunakan metode Spill Forest. Objek penelitian mereka beragam yaitu *foto*, panorama *picture, image satellite*, dan *vector image.*

Charles Norona, Tyagi dan Vivek Kumar [7] di dalam papernya yang berjudul *Non-distributed Object Recognition Potential on the Android Platform* menerapkan proses *recognition* pada *mobile device* dengan platform android menggunakan metode *SURF*. Dari penelitian ini mereka membuktikan bahwa *SURF* lebih cepat melakukan komputasi proses *feature detection* dan *feature descriptor* pada platform android. Objek penelitian mereka adalah *image photographic* yang diambil secara langsung dari kamera.

Mosalam Ebrahimi dan Walterio W. Mayol-Cuevas [8] di dalam papernya yang berjudul *Adaptive Sampling for Feature Detection, Tracking and Recognition on Mobile Platforms* memodifikasi metode *FAST* (*Features from Accelerated Segment Test*) untuk diimplementasikan ke dalam *handphone*. Hasil

modifikasi dari metode *FAST* adalah *M-FAST* (*Mobile Features from Accelerated Segment Test*). Selain itu mereka juga memodifikasi *BRIEF* untuk membentuk *adaptive descriptor*. Mereka menyatakan bahwa metode *M-FAST* yang mereka temukan lebih cepat dari metode *SURF*. Sedangkan untuk proses *feature matching* mereka menggunakan metode *FLANN TREE*. Objek penelitian yang mereka teliti adalah video atau gambar bergerak.

Sami Mohammad Halawani, Dzulkifli Mohammad [9] di dalam papernya yang berjudul *Logo Matching Technique Based on Principle Component Analysis* melakukan penelitian terhadap logo dengan metode *principle component analysis.*

Dari *state of the art* tentang proses *recognition* maka tingkat akurasi dari penelitian yang berhubungan dengan *vector image* masih beragam meskipun tidak sepenuhnya menggunakan ketiga tahapan proses *recognition*. Sehingga belum ditemukan metode yang akurat untuk melakukan proses recognition dengan menggunakan ketiga tahapan proses *recognition* dengan metode yang berbeda di setiap tahapannya.

1.2. Rumusan Masalah

a. Umum
   Proses *recognition* pada logo sebagai *vector image* sulit dilakukan jika ada perbedaan kondisi tampilan seperti *scale invariant* dan *rotation invariant.*
b. Spesifik
   Belum ditemukannya metode yang tepat dan tingkat akurasi yang tinggi dalam proses *feature recognition* pada logo berbasis *corner detection* dalam *augmented reality system* melalui tahapan *feature extraction, feature descriptor,* dan *feature matching*.

1.3. Tujuan Penelitian

a. Umum
   Penelitian ini bertujuan menemukan metode yang tepat untuk mengatasi masalah dalam proses *recognition* logo pada kondisi *scale invariant* dan *rotation invariant.*
b. Spesifik
   Meningkatkan akurasi pada proses *feature* recognition logo berbasis corner detection melalui tahapan *feature* extraction, *feature* descriptor, dan *feature matching* dengan metode FAST, SURF, dan FLANN TREE.

1.4. Manfaat

Manfaat dari penelitian ini adalah :

a. Manfaat bagi masyarakat atau perusahaan
   Penelitian ini diharapkan dapat digunakan untuk pengembangan aplikasi *augmented reality* dengan metode FAST, SURF, dan FLANN TREE.
b. Manfaat bagi Iptek
   Hasil penelitian ini diharapkan dapat *member*ikan sumbangan bagi pengembangan teori yang berkaitan dengan proses *feature recognition* berbasis *corner detection* menggunakan metode FAST, SURF, dan FLANN TREE untuk pengembangan sistem *augmented reality*.

2. TINJAUAN PUSTAKA

2.1. Penelitian yang Relevan

Penelitian berbagai macam metode untuk membentuk sistem *augmented reality* dibagi menjadi dua bagian yaitu:

a. Berbasis *Feature*
b. Berbasis *Training*

Pada pembuatan sistem *augmented reality* berbasis *feature* maka proses yang dikerjakan ada tiga bagian yaitu:

a. Pembentukan *Feature Extractor* (Proses *Feature* Detection dan Extraction)
   Memiliki metode yang berbasis *corner* dan berbasis *region.*
b. Komputasi *Feature Descriptor*
   Memiliki metode berbasis distribution dan berbasis filter.
   Metode berbasis distribution antara lain metode Sift, Surf dan Brief
c. Proses *Feature Matching*
   Memiliki beberapa metode seperti *KD Tree, Spill Tree* dan *Flann Tree.*

Sedangkan pada pembuatan sistem *augmented reality* berbasis *training* memiliki metode berbasis klasifikasi seperti *Ferns* dan *Oneway descriptor*.

Penelitian ini mengacu pada penelitian sebelumnya yaitu penelitian oleh Mosalam Ebrahimi dan Walterio W. Mayol-Cuevas [8] dan penelitian oleh Sami Mohammad Halawani, Dzulkifli Mohammad [9]. Yang membedakan penelitian ini dari penelitian Mosalam Ebrahimi dan Walterio W. Mayol-Cuevas adalah penggunaan metode yang berbeda pada proses *feature descriptor*. Persamaan penelitian ini dengan penelitian Mosalam Ebrahimi dan Walterio W. Mayol-Cuevas adalah menggunakan metode *Fast* dan *Flann Tree*. Persamaan penelitian ini dengan penelitian Sami Mohammad Halawani, Dzulkifli Mohammad adalah penggunaan objek penelitian yang sama yaitu logo. Penelitian ini menggunakan metode berbasis *feature* khususnya pada bagian *feature descriptor* menggunakan metode *Surf* seperti pada penelitian Wei-Chao Chen, Yingen Xiong, Jiang Gao, Natasha Gelfand, dan Radek Grzeszczuk [2], Gabriel Takacs [4], dan Charles Norona, Tyagi dan Vivek Kumar [7]. Sedangkan pada proses pembentukan *feature extractor* berbasis *corner* menggunakan metode *Fast* seperti pada penelitian Niels Henze, Torben Schinke dan Susanne Boll [5], Daniel Wagner [6], dan Mosalam Ebrahimi dan Walterio W. Mayol-Cuevas [8].

2.2. Landasan Teori

2.2.1 *Feature Recognition*

Feature recognition adalah proses untuk mengidentifikasi *image* menggunakan metode berbasis *feature*. Berbasis *feature* artinya mengutamakan bagian – bagian dari *local feature* dalam sebuah *image* seperti letak *corners, edges* dan *lines.* Tahap – tahap di dalam proses *feature recognition* terbagi ke dalam tiga tahap yaitu:

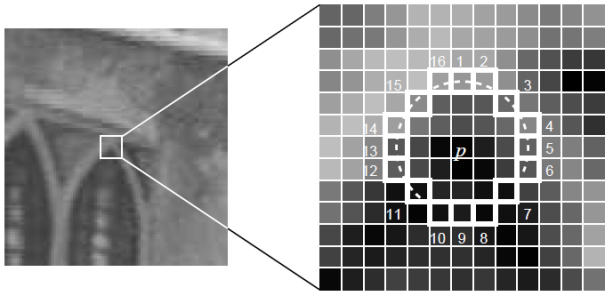
a. *Feature detection and extraction*
b. *Feature descriptor computation*
c. *Feature matching*

Feature detection adalah metode untuk mendeteksi *corner* yang terdapat dalam sebuah *image.* Pada tahap ini, sistem seharusnya memiliki kemampuan yang cepat dan akurat untuk menemukan banyak *corner* dalam kondisi *image* yang bervariasi. Tahap *feature detection* akan menghasilkan *detector*. Setelah *corner* dideteksi kemudian pada tahap *feature extraction*, kuantitas *corner* dihitung agar dapat dibandingkan dan dianalisis. Pada tahap *feature extraction* akan terbentuk *feature descriptor* atau *descriptor vector*. *Feature detection* dan *feature extraction* sangat menentukan untuk mencari korespodensi antara lokal *image* yang kondisinya kurang baik. Untuk penelitian ini *detector* yang digunakan adalah *FAST* yang akan mendeteksi *corner* dengan cepat sedangkan *SURF* akan digunakan sebagai *detector* sekaligus *descriptor* untuk mendeteksi dan mengekstraksi *corner* yang memiliki variasi perbedaan pada *image* seperti perbedaan skala dan rotasi.

Tahap *descriptor computation* adalah proses komputasi untuk membentuk *SURF descriptor*.

Tahap *feature matching* adalah tahap mencocokkan atau membandingkan *corner* yang telah diekstraksi dengan *image data class* yang tersimpan dalam *database* untuk menemukan korespondensi dan relasi yang cocok dan terbaik pada *image*. Jika tahap *feature matching* berhasil dengan baik maka *image* dapat dikenali oleh sistem.

2.2.2 Feature Detection dan Extraction Berbasis Features from Accelerated Segment Test-FAST [10]
Features from Accelerated Segment Test (FAST) merupakan metode untuk mendeteksi *corner* dengan kemampuan deteksi kecepatan tinggi (*high-speed detection*). Metode ini akan menentukan *segment test*. *Segment test* adalah potongan *image* yang diambil, yang di dalamnya memiliki *corner* yang akan dideteksi. Cara untuk mempercepat deteksi *corner* dalam metode *FAST* adalah dengan menentukan sebuah lingkaran yang terdiri dari enam belas *pixel* yang mengelilingi *p* sebagai *corner* atau sudut yang akan dideteksi. *p* akan diklasifikasikan sebagai *corner* jika dalam lingkaran ada sekelompok *n pixel* yang berdekatan. Sekelompok *n pixel* tersebut semuanya memiliki intensitas yang lebih terang daripada intensitas *pixel* I_p ditambah dengan ambang batas *t* (*threshold*) $(I_p + t)$. Atau semuanya memiliki intensitas lebih gelap daripada I_p dikurangi dengan t $(I_p - t)$. Seperti ilustrasi yang ditampilkan pada gambar 1.



Gambar 1. Dua Belas *Point Segment Test* pada *Corner* [15]

Pada gambar 1, persegi – persegi yang ditandai dengan garis terang adalah *pixel – pixel* yang digunakan untuk mendeteksi *corner*. *Pixel p* sebagai pusat adalah *corner* yang dideteksi. Garis lengkung yang ditandai dengan garis putus-putus adalah *pixel* berjumlah dua belas yang saling berdekatan yang memiliki intensitas lebih terang dari *p* didefinisikan dengan rumus $(I_p + t)$. Jumlah *pixel* yang berdekatan dengan *p* ditetapkan dengan *n* sebanyak dua belas *pixel*. *n* ditetapkan sebanyak dua belas karena dengan jumlah tersebut sistem mampu melakukan proses pendeteksian *pixel* dengan sangat cepat dan digunakan untuk memisahkan banyak *non corner*. Di antara kedua belas *pixel* yang ditentukan untuk dideteksi, pendeteksian *pixel* akan menguji empat *pixel* yaitu *pixel – pixel* yang terletak di urutan ke 1, 5, 9 dan 13 (searah dengan empat penjuru kompas). Jika *p* dideteksi sebagai *corner* maka paling sedikit ada tiga *pixel* yang memiliki intensitas lebih terang dari $(I_p + t)$ atau lebih gelap dari $(I_p - t)$. Jika tidak memenuhi ketentuan tersebut maka *p* bukan *corner*. Setiap *pixel* dalam lingkaran diuji dengan cara yang sama untuk menentukan *corner p*. Dari cara tersebut metode *FAST* yang asli memiliki kelemahan yaitu:

a. *High-speed test* tidak dapat bekerja dengan baik jika $n < 12$
b. Kemunculan penyebaran *feature* yang dideteksi tidak tampak dalam sistem
c. Hasil pendeteksian empat *pixel* sebelumnya akan terbuang
d. *Feature* yang dideteksi letaknya saling berdekatan satu dengan yang lain sehingga menimbulkan pendeteksian *corner* yang sama lebih dari satu.

Untuk memperbaiki kelemahan pada metode *FAST* yang asli maka dalam metode *FAST* perlu ditambahkan *machine learning* untuk menentukan tiga jenis intensitas dari *pixel* yaitu *darker*, *similar* dan *brighter*. Proses penentuan ini terdiri dari dua tahap. Tahap pertama, untuk menentukan *corner detector* yang berjumlah *n* maka *corner* dideteksi dari *image* target menggunakan *segment test* dengan enam belas *pixel* yang membentuk lingkaran seperti yang telah dijelaskan sebelumnya. Pada setiap lokasi dalam lingkaran, ditentukan $x \in \{1..16\}$, *x* adalah *pixel* yang memiliki posisi relatif terhadap *p* (ditandai

dengan rumus $p \to x$). *x* dapat memiliki salah satu dari tiga keadaan (states) sebagai berikut:

$$S_{p\to x} = \begin{cases} d, & I_{p\to x} \le I_p - t \ (darker) \\ s, & I_p - t < I_{p\to x} < I_p + t \ (similar) \\ b, & I_p + t \le I_{p\to x} \ \ (brighter) \end{cases} \quad (1)$$

Setelah *x* ditentukan kemudian $S_{p\to x}$ dihitung untuk setiap $p \in P$ (*P* adalah semua *pixel* dalam *image* yang di*training*). *P* dikelompokkan ke dalam tiga subset yaitu P_d, P_s, P_b dan setiap *p* dikelompokkan ke dalam subset $P_{S_{p\to x}}$. Kemudian variabel *boolean* K_p ditentukan akan bernilai benar (true) jika *p* adalah *corner* dan akan bernilai salah *(false)* jika bukan *corner (non corner)*.

Tahap kedua, dengan pendekatan menggunakan metode *induction of decision tree* (ID3) dan dimulai dengan memilih *x* yang menghasilkan informasi apakah sebuah *pixel* adalah *corner*, dihitung dengan *entropy* dari K_p.

Entropy dari *K* untuk *set* P adalah:

$$H(P) = (c + \bar{c}) \log_2 (c + \bar{c}) - c \log_2 c - \bar{c} \log_2 \bar{c} \quad (2)$$

Keterangan:

$c = |\{p | K_p \text{ is true}\}|$ (jumlah *corners*)

dan $\bar{c} = |\{p | K_p \text{ is false}\}|$ (jumlah *non corners*)

Pemilihan *x* akan menghasilkan informasi mengenai jumlah *corners* dan *non corners*:

$$H(P) - H(P_d) - H(P_s) - H(P_b) \quad (3)$$

Sesudah memilih *x*, kemudian proses diaplikasikan secara rekursif pada ketiga subset, contohnya x_b dipilih untuk subset P_b pada $P_{b,d}$, $P_{b,s}$, $P_{b,b}$ kemudian x_s dipilih untuk subset P_s pada $P_{s,d}$, $P_{s,s}$, $P_{s,b}$ dan x_d dipilih untuk subset P_d pada $P_{d,d}$, $P_{d,s}$, $P_{d,b}$, saat x dipilih maka informasi mengenai *set* juga diproses. Proses ID3 akan berhenti jika *entropy* dari *subset* adalah nol. Jadi semua p dalam setiap *subset* P memiliki nilai K_p yang sama, untuk *corners* atau *non corners*.

Untuk mengatasi kelemahan yang keempat pada *FAST* yang asli, maka metode *non-maximal suppression* diterapkan. Nilai maksimum V diaplikasikan pada setiap *corner* yang dideteksi dan *non-maximal suppression* diterapkan untuk menghapus setiap *corner* yang berdekatan yang nilainya lebih tinggi dari V.

Penerapan nilai maksimum V dapat dipahami dengan beberapa definisi yaitu:

a. Nilai maksimum V diperoleh dari n di mana p merupakan *corner*.
b. Nilai maksimum V diperoleh dari t di mana p merupakan *corner*.
c. Nilai maksimum V akan menjumlahkan perbedaan yang absolut (mutlak) antara *pixel* yang bersinggungan pada busur dengan *pixel* pusat yang didefinisikan sebagai berikut:

$$V = \max\left(\sum_{x \in S_{bright}} |I_{p\to x} - I_p| - t \ , \quad \sum_{x \in S_{dark}} |I_p - I_{p\to x}| - t\right) \quad (4)$$

Dengan definisi dari S_{bright} dan S_{dark} sebagai berikut:

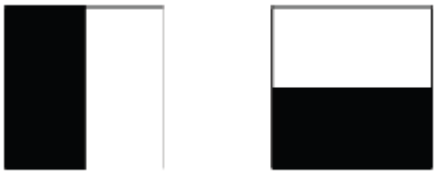
$$S_{bright} = \{x | I_{p\to x} \ge I_p + t\}$$
$$S_{dark} = \{x | I_{p\to x} \le I_p - t\}$$

2.2.3 *Feature Descriptor* Berbasis *Speeded-Up Robust Feature* (SURF) [3]

Di dalam metode *SURF* akan terbentuk *detector* dan *descriptor* yang digunakan untuk mendeteksi dan mengekstraksi *corners* dalam sebuah *image*. Metode *SURF* dapat melakukan deteksi dan ekstraksi pada *patch image* atau potongan *image* yang memiliki perbedaan skala dan rotasi.

SURF Descriptor

SURF descriptor berguna untuk menguraikan penyebaran intensitas *pixel* dengan skala tertentu di sekeliling *image* untuk setiap *corners* yang telah dideteksi oleh *FAST-Hessian*. *Filter*isasi dengan integral *image* di dalam *SURF* dikenal dengan sebutan *Haar Wavelet*. Pem*filter*an dengan *Haar Wavelet* bertujuan untuk meningkatkan kekokohan pada *descriptor* dan mengurangi waktu komputasi. *Haar Wavelet* juga merupakan *filter* yang digunakan untuk menemukan gradien pada arah x dan y, seperti yang ditunjukkan oleh gambar 2.



Gambar 2. *Haar Wavelet*

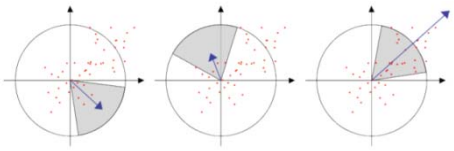
Pada gambar 2 bagian kiri, Respon dari *Haar Wavelet* pada arah y dan gambar 2 bagian kanan adalah respon dari *Haar Wavelet* pada arah x.

Proses ekstraksi pada *SURF descriptor* dibagi ke dalam dua tugas yang berbeda yaitu:

a. Memperbaiki orientasi reproduksivitas *corner* yang berdasarkan informasi dari area lingkaran di sekeliling *corner*.
b. Membentuk jendela *descriptor* untuk mengekstrak vektor *SURF descriptor* berdimensi 64. Jendela *descriptor* memiliki kotak masukan bagi *pixel – pixel descriptor*. Arah kotak sejajar dengan orientasi yang dipilih.

Penentuan Arah Orientasi *Descriptor*

Untuk mengatasi perbedaan arah rotasi pada *image* maka setiap *corner* yang dideteksi akan ditentukan arah orientasinya. Arah orientasi ini akan menentukan arah vektor *descriptor* yang akan diekstraksi dan hal ini sangat berpengaruh terhadap banyaknya variasi perbedaan rotasi pada sebuah *image*. Untuk menentukan arah orientasi maka perlu dihitung respon dari *Haar Wavelet* pada arah x dan y. Panjang dari *Haar Wavelet* adalah 4σ. σ adalah skala saat *corner* dideteksi. *Haar Wavelet* akan dihitung pada radius 6σ dari corner yang dideteksi. Respon dari *Haar Wavelet* di pusat corner akan diberi bobot oleh Gaussian sebesar 2.5σ. Arah respon dari *Haar Wavelet* direpresentasikan sebagai vektor dengan arah respon horisontal jika respon lebih banyak sepanjang garis absis (x) dan arah respon vertikal jika respon lebih banyak sepanjang garis ordinat (y). Arah orientasi dominan diperkirakan dengan cara menghitung jumlah semua respon pada area dengan sudut $\frac{\pi}{3}$. Respon dengan arah vertikal dan horisontal dijumlahkan untuk membentuk vektor yang baru. Arah orientasi dari *corner* yang dideteksi ditentukan oleh nilai vektor yang terpanjang.



Gambar 3. Penentuan arah orientasi *descriptor* [16]

Pada gambar 3, penentuan arah orientasi *descriptor* ditandai dengan garis panah biru dan area orientasi dominan ditandai dengan area berwarna biru.

Descriptor Components

Langkah awal untuk mengektraksi *SURF descriptor* adalah membentuk jendela persegi yang berpusat di sekitar *corner* dan memiliki orientasi arah yang telah ditentukan sebelumnya. Ukuran jendela ini adalah 20σ. Jendela ini akan memuat *pixel – pixel* yang akan menjadi vektor *descriptor*.

Gambar 4. Jendela Persegi yang Terbentuk pada *SURF Descriptor* [3]

Pada gambar 4, jendela persegi yang terbentuk berukuran 20σ. Arah orientasi yang lebih dominan ditandai dengan warna hijau.

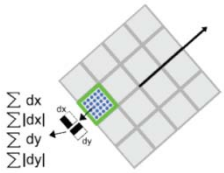
Jendela ini dibagi lagi menjadi kotak persegi yang lebih teratur dengan ukuran yang lebih kecil yaitu 4x4 dan disebut dengan nama *subregion*. *Haar Wavelet* dengan ukuran 2σ akan dihitung pada setiap *subregion* dan menghasilkan 25 *sample point* yang tersebar secara teratur.

Respon dari *Haar Wavelet* yang berada pada arah horisontal ditandai dengan d_x dan respon dari *Haar Wavelet* yang berada pada arah vertikal ditandai dengan d_y. Sedangkan jika ada polaritas (dua sifat yang berlawanan) dari perubahan intensitas pada *pixel* maka respon ini memiliki nilai absolut. Respon pada arah horisontal dan vertikal yang mempunyai nilai absolut ditandai dengan $|d_x|$ dan $|d_y|$.

Setiap *subregion* mempunyai empat dimensi vektor *descriptor* (v) yang dirumuskan dengan:

$$v_{subregion} = [\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|] \quad (5)$$

Jadi keseluruhan vektor *descriptor* pada *subregion* memiliki panjang 4x4x4=64



Gambar 5. Komponen dari *Descriptor* [16].

Pada gambar 5, kotak yang berwarna hijau adalah salah satu contoh *subregion* dari 16 *subregion* yang

dibentuk. Bulatan berwarna biru adalah *sample point* yang merupakan hasil perhitungan atau respon dari *Haar Wavelet*.

2.2.4 *Feature Matching* Berbasis FLANN TREE [4]

Metode *FAST Library for Approximate Nearest Neighbors* (FLANN) *Tree* bertujuan untuk melakukan proses *matching* pada *image* dengan cara mencari estimasi *nearest neighbor* dari *corner image* dalam *dataset*. Keakuratan metode FLANN di ukur dengan kecepatan untuk menemukan estimasi *nearest neighbor* dan persentase *nearest neighbor* yang tepat sesuai permintaan *point query*. Metode yang optimal untuk mencari estimasi *nearest neighbor* sangat bergantung pada beberapa faktor yaitu:

a. Struktur *dataset* (adanya korelasi antara *feature* dalam *dataset*)
b. Akurasi pencarian yang diinginkan

Metode FLANN akan mengoptimalkan urutan tingkatan (hirarki) dalam metode *multiple randomized kd-tree* dan *k-means tree*.

Secara tradisional, setiap metode memiliki sekelompok parameter yang *member*ikan pengaruh signifikan pada proses pencarian data. Parameter ini dapat berupa sejumlah *randomized trees* yang terdapat pada metode *kd-tree* atau cabang- cabang dan sejumlah perulangan (iterasi) yang terdapat pada metode hirarki *k-means tree*. Metode FLANN bekerja dengan cara memperkecil parameter untuk mengoptimalkan pencarian *nearest neighbor* dalam *dataset*. Parameter yang digunakan dalam metode FLANN yaitu parameter untuk *search time, tree build time*, dan penggunaan *tree memory*. Berdasarkan metode FLANN maka pembatasan penggunaan *memory* dan waktu dapat diatur. Jika sistem hanya membutuhkan untuk membangun *tree* sekali saja dan memanfaatkannya untuk permintaan *query* data yang besar, maka *tree build time* dapat diabaikan. Pada kasus yang lain, *tree build time* dan *search time* dapat diatur lebih kecil jika *tree* dibangun secara online dan proses pencarian dikerjakan lebih cepat.

Rumus *cost* akan diterapkan untuk menghitung *search time, tree build time*, dan *tree memory overhead*.

$$cost = \frac{s + w_b B}{(s + w_b b)_{opt}} + w_m m \qquad (6)$$

Penjelasan dari rumus tersebut adalah:

s adalah waktu pencarian (*search time*) sejumlah vektor pada contoh *dataset*.

B adalah waktu untuk membentuk *tree* (*tree build time*).

$m = {}^{m_t}/_{m_d}$, *m* adalah rasio perbandingan penggunaan memory untuk tree (m_t) dan penyimpanan data (m_d).

w_b adalah bobot untuk membentuk *tree* (*build-time weight*) yang akan mengendalikan *tree build time* relatif dengan waktu pencarian (*search time)*

$w_b = 0$ adalah waktu pencarian tercepat. Jika $w_b = 1$ maka *tree build time* dan *search time* memiliki kepentingan yang sama.

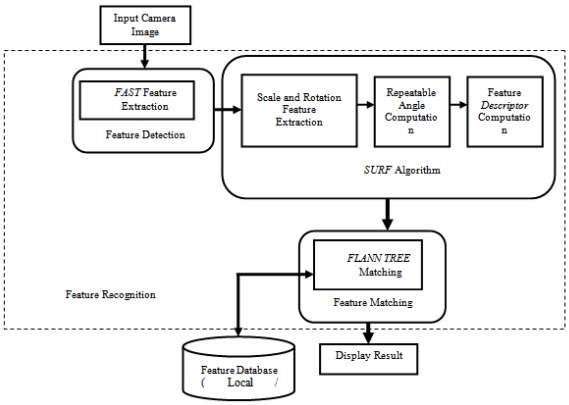
w_m adalah *memory weight* yang akan mengendalikan kelebihan *memory* berbanding dengan kelebihan waktu.

Kelebihan waktu dihitung secara relatif pada *time cost* $(s + w_b b)_{opt}$ optimal. Jika $w_m = 0$ akan menghasilkan pencarian tercepat tanpa memperhatikan kelebihan memory. Jika disetting $w_m = 1$ maka akan *member*ikan bobot yang setara pada persentase kenaikan memory yang digunakan sama dengan persentase kenaikan pada *search time* dan *build time*.

2.3. Model *Feature Recognition* Berbasis *Corner Detection* dengan Metode FAST, SURF, dan FLANN TREE

Gambar 6 adalah ilustrasi proses dari *feature recognition* berbasis *corner detection* dengan metode *FAST, SURF, dan FLANN TREE*. Logo akan di*capture* melalui kamera. Ketika logo di*capture* akan ada tiga

kondisi yang diberikan yaitu normal, perbedaan skala (*scale invariant*), dan perbedaan rotasi (*rotation invariant*). Kemudian data dari logo di*capture* akan melalui proses *corner detection* dan *extraction* dengan menggunakan metode Fast. Setelah diperoleh letak *corner* maka proses selanjutnya adalah membentuk *descriptor* dengan metode Surf. Dari pembentukan *descriptor* ini akan diperoleh data yang akan menunjukkan letak *corner* pada logo yang di*capture* dengan data logo di dalam *database*. Kemudian proses *matching* bisa dilakukan dengan metode Flann Tree.



Gambar 6. Kerangka Teori Proses *Feature Recognition*

Yang terdiri dari:

a. *Input camera* untuk mengambil *image* dari kamera perangkat *mobile*.
b. *Feature detection* dilakukan dengan metode *FAST*
c. *Feature descriptor* dengan metode *SURF*
d. *Feature matching* menggunakan metode *FLANN TREE*
e. Dan *display result* akan menampilkan hasil *feature matching* dari pengambilan *image* yang berupa logo dengan logo di dalam *database* lokal.

3. METODE PENELITIAN

3.1. Metode Pengumpulan Data

Penelitian ini menggunakan data primer yang diperoleh dari hasil eksperimen terhadap *corner* pada logo. Data primer dalam penelitian ini adalah data yang dihasilkan dari perhitungan waktu komputasi pada parameter yang telah ditentukan. Selain itu juga diperoleh data perbandingan akurasi proses *recognition* antar *approach* yang ditentukan menggunakan metode FAST,SURF, FLANN TREE; metode FAST, SIFT, FLANN TREE serta metode FAST, BRIEF, FLANN TREE.

3.2. Metode Pengolahan Awal Data

Pengolahan awal data dilakukan terhadap data berbentuk *image* yang berjenis logo sebanyak 50 jenis logo. Pengolahan awal data berbentuk logo ini terdiri dari beberapa tahap, yaitu:

a. Menentukan resolusi untuk setiap logo yaitu dengan resolusi maksimal 320 x 320 pixel.
b. Data yang berupa data sumber adalah data yang bersifat normal artinya tidak ada perlakuan khusus seperti perbedaan skala, perbedaan rotasi dan memiliki kualitas warna standar.
c. Ketika data logo sebagai data sumber di*input* ke dalam sistem maka warna pada logo akan diubah menjadi warna *grayscale*.
d. Data kedua akan di*input* melalui kamera pada sistem, data ini akan ditangkap oleh kamera sesuai dengan warna aslinya.
e. Saat data kedua ditangkap oleh kamera, selain logo diperlakukan pada kondisi normal, ada kondisi khusus yang diberikan pada logo yaitu adanya perbedaan skala dan rotasi sebesar 90^0 dan 180^0 pada data logo.

f. Kemudian sistem akan melakukan proses identifikasi logo pada data sumber dan data yang di*input* melalui kamera.

3.3. Eksperimen

Tahapan proses untuk melakukan eksperimen logo *recognition* sebagai berikut:

Tahap 1: Proses *feature detection* dan *extraction* dengan metode FAST

a. Penentuan intensitas p*ixel*
   1) Pada tahap *feature detection* ditentukan *segment test* pada masing – masing logo dari 50 logo yang merupakan objek penelitian.
   2) Intensitas pixel dari *segment test* yang diambil ditentukan dengan rumus (1).
   3) Pada rumus (1) dihitung untuk setiap pixel $p \in P$ (*P* adalah semua *pixel* dalam *image* yang di*training*).
   4) P dikelompokkan ke dalam tiga subset yaitu P_d, P_s, P_b dan setiap pixel *p* dikelompokkan ke dalam subset $P_{S_{p \to x}}$.
   5) Penentuan dan perhitungan c*orner*
   6) Setelah pixel *p* dikelompokkan ke dalam subset $P_{S_{p \to x}}$ kemudian variabel *boolean* K_p ditentukan bernilai benar (true) jika *p* adalah *corner* dan bernilai salah *(false)* jika bukan *corner (non corner)*.
   7) Untuk menentukan sebuah *pixel* merupakan *corner* dihitung dengan *entropy* dari K_p pada rumus (2).
   8) Jumlah *corner* dan *non corner* dihitung dengan rumus (3)
b. Pendeteksian *corner* yang berdekatan dengan n*on-maximal suppression*
   1) Menentukan *non-maximal suppression* dengan mencari nilai maksimum V untuk setiap *corner* yang berdekatan berdasarkan rumus (4).
   2) Jika nilainya lebih tinggi dari V*max* maka setiap *corner* yang berdekatan akan dihapus.

Tahap 2: Proses *feature descriptor* pada metode SURF

a. Pemfilteran corner dengan *Haar Wavelet*.
   1) Menentukan gradien pada arah dx, dy, |dx| dan |dy|.
b. Menentukan vektor *descriptor* SURF berdasarkan rumus (5).

Tahap 3: Proses *feature matching* dengan metode FLANN TREE berdasarkan rumus (6)

a. Penentuan index pada pencarian *nearest neighbor* dari *corner* logo sumber dan *corner* logo *capture* berdasarkan parameter dari *KD Tree Index*.
b. Menentukan korespondensi dengan NNDR (*Nearest Neighbor Distance Ratio).*
c. Menemukan *homography* berdasarkan *inlier* dan *outlier* dari logo sumber dan logo *capture*.

3.4. Pengujian

Pengujian dilakukan dengan mengacu pada beberapa kondisi yaitu kondisi *normal*, *scale invariant*, *rotation* 90^0 dan *rotation* 180^0. Pada kondisi normal logo yang di*capture* memiliki kondisi yang sama dengan data logo pada *database* sumber. Sedangkan pada kondisi *scale invariant*, logo yang dicapture memiliki perbedaan skala dengan data logo sumber. Selain itu pada kondisi *scale invariant*, logo yang *dicapture* memiliki *struktur* image yang berbeda dengan logo pada data sumber. Pada kondisi *rotation* 90^0 dan *rotation* $180^{0,}$ logo yang di*capture* akan diputar dengan sudut 90^0 dan $180^{0.}$

Metode yang diuji adalah metode *FAST, SURF, dan FLANN TREE*; metode *Fast*, *Sift* dan *Flann Tree* dan metode *Fast*, *Brief* dan *Flann Tree*. Dari hasil pengujian didapatkan akurasi dari metode *Fast,Surf* dan *Flann Tree* lebih tinggi dibandingkan dengan metode *Fast*, *Sift* dan *Flann Tree* serta *Fast*, *Brief* dan *Flann Tree.*

Pengujian akan menghitung kecepatan waktu komputasi dari beberapa parameter yang digunakan

sebagai acuan yaitu:

a. *Feature detection* (ms): digunakan untuk menghitung lama waktu dilakukannya proses *corner detection.*
b. *Descriptor extraction* (ms): digunakan untuk menghitung lama waktu dilakukannya proses ekstraksi *descriptor* untuk menentukan arah vektor *descriptor*.
c. *Descriptor indexing* (ms): digunakan untuk menghitung lama waktu dilakukannya proses *indexing* pada *descriptor* yaitu *member*ikan nilai *index* pada letak vektor *descriptor*.
d. *Descriptor matching* (ms): digunakan untuk menghitung lama waktu dilakukannya proses *descriptor matching* pada data sumber dan data yang diambil melalui kamera.
e. *Detect outlier dan GUI* (ms): digunakan untuk menghitung lama waktu dilakukannya proses mendeteksi *outline* dan GUI dari logo.

Pengujian dengan parameter untuk menentukan akurasi area *corner* yang ditemukan yaitu:

a. *Min matched distance*: digunakan untuk menghitung banyaknya area minimal yang sama antara data sumber dan data yang diambil melalui kamera .
b. *Max matched distance*: digunakan untuk menghitung banyaknya area maksimal yang sama antara data sumber dan data yang diambil melalui kamera.
c. *Corner quantity*: digunakan untuk menghitung banyaknya jumlah *corner*.

4. HASIL PENELITIAN DAN PEMBAHASAN

4.1. Hasil Eksperimen dan Pengujian

4.1.1 Hasil Eksperimen dengan Metode FAST, SURF, dan FLANN TREE

Tabel 1. Rata-Rata Kecepatan Komputasi pada Metode FAST, SURF, dan FLANN TREE

	Feature Detection (ms)				Descriptor Extraction (ms)				Descriptor Indexing (ms)				Descriptor Matching (ms)				Detect Outlier and GUI (ms)			
Rata - Rata Kecepatan Komputasi	3.92	3.9	4	4.28	44.5	38.26	36.22	53.24	6.86	5.72	5.4	8.34	8.5	8.44	8.68	8.62	81.8	86.38	83.92	79.14

4.1.2 Hasil Eksperimen dengan Metode Fast, Brief dan Flann Tree

Tabel 2. Rata-Rata Kecepatan Komputasi pada Metode Fast, Brief dan Flann Tree

	Feature Detection (ms)		Descriptor Extraction (ms)		Descriptor Indexing (ms)		Descriptor Matching (ms)		Detect Outlier and GUI (ms)	
Rata - Rata Kecepatan Komputasi	3.84	3.74	5.74	5.42	3.66	3.34	5.7	5.56	63.84	64.74

4.1.3 Hasil Eksperimen dengan Metode Fast, Sift dan Flann Tree

Tabel 3. Rata-Rata Kecepatan Komputasi pada Metode Fast, Sift dan Flann Tree

	Feature Detection (ms)		Descriptor Extraction (ms)		Descriptor Indexing (ms)		Descriptor Matching (ms)		Detect Outlier and GUI (ms)	
Rata - Rata Kecepatan	3.84	3.66	221.82	202.24	10.36	8.78	10.68	10.58	96.58	78.88

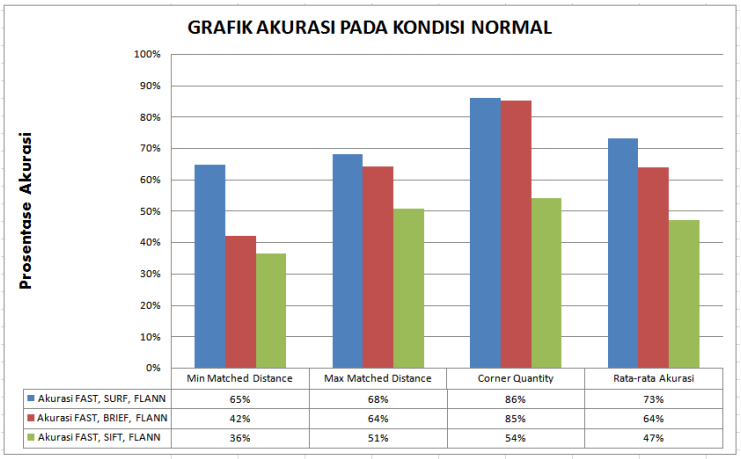
Tabel 1,2, dan 3 memperlihatkan rata-rata kecepatan komputasi dengan parameter *feature* detection, descriptor extraction, descriptor indexing, descriptor *matching* dan detect outlier and GUI.

4.1.4 Hasil Perbandingan Akurasi pada Kondisi Normal

Tabel 4. Perbandingan Akurasi pada Kondisi Normal

PERHITUNGAN / METODE	Min Matched Distance	Max Matched Distance	Corner Quantity	Rata-rata Akurasi
Akurasi FAST, SURF, FLANN	65%	68%	86%	**73%**
Akurasi FAST, BRIEF, FLANN	42%	64%	85%	**64%**
Akurasi FAST, SIFT, FLANN	36%	51%	54%	**47%**

Dari Tabel 4 terlihat bahwa akurasi metode FAST, SURF, dan FLANN TREE memiliki tingkat rata – rata akurasi 9% lebih besar dibandingkan metode Fast, Brief dan Flann Tree dan 26% lebih besar dibandingkan metode Fast, Sift dan Flann Tree.



Gambar 7. Grafik Akurasi pada Kondisi Normal

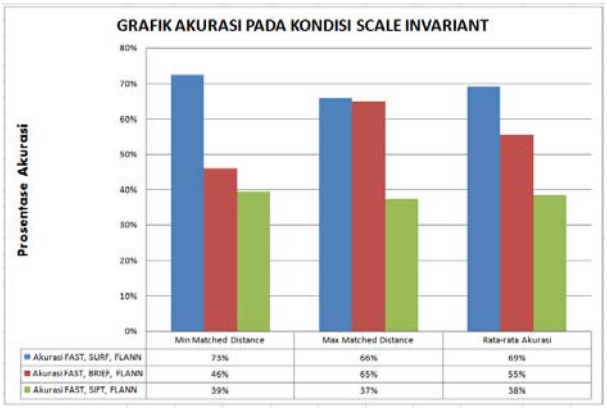
Dari gambar 7 diketahui bahwa akurasi metode FAST, SURF, dan FLANN TREE lebih tinggi dibandingkan dengan metode lainnya, yang ditunjukkan oleh bar berwarna biru.

4.1.5 Hasil Perbandingan Akurasi pada Kondisi Scale Invariant

Tabel 5. Perbandingan Akurasi pada Kondisi *Scale Invariant*

PERHITUNGAN / METODE	Min Matched Distance	Max Matched Distance	Rata-rata Akurasi
Akurasi FAST, SURF, FLANN	73%	66%	**69%**
Akurasi FAST, BRIEF, FLANN	46%	65%	**55%**
Akurasi FAST, SIFT, FLANN	39%	37%	**38%**

Dari Tabel 5 terlihat bahwa akurasi metode FAST, SURF, dan FLANN TREE memiliki tingkat rata – rata akurasi 14% lebih tinggi dibandingkan metode Fast, Brief dan Flann Tree dan 31% lebih tinggi dibandingkan metode Fast, Sift dan Flann Tree.



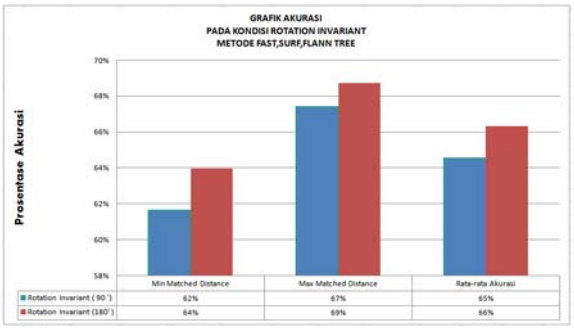
Gambar 8. Grafik Akurasi pada Kondisi *Scale Invariant*

Dari gambar 8 diketahui bahwa akurasi metode FAST, SURF, dan FLANN TREE pada kondisi *scale invariant* lebih tinggi dibandingkan dengan metode lainnya, yang ditunjukkan oleh bar berwarna biru.

Tabel 6. Akurasi pada Kondisi *Rotation Invariant*

METODE FAST, SURF, FLANN TREE	**Min Matched Distance**	**Max Matched Distance**	**Rata-rata Akurasi**
Rotation Invariant (90 ')	62%	67%	**65%**
Rotation Invariant (180')	64%	69%	**66%**

Dari Tabel 6 terlihat bahwa akurasi metode FAST, SURF, dan FLANN TREE memiliki tingkat rata – rata akurasi 65% pada sudut 90^0 dan 66% pada sudut 180^0.



Gambar 9. Grafik Akurasi pada Kondisi *Rotation Invariant*

Dari gambar 9 diketahui bahwa memiliki rata-rata akurasi metode FAST, SURF, dan FLANN TREE pada kondisi *rotation invariant* 180^0 lebih tinggi dibandingkan dengan pada kondisi *rotation invariant* 90^0, yang ditunjukkan oleh bar berwarna merah.

4.2. Implikasi Penelitian

Dari hasil penelitian yang telah dilakukan, tingkat akurasi pendeteksian logo berbasis *corner detection* sangat bergantung pada jumlah atau kuantitas *corner* yang dideteksi pada logo. Semakin banyak *corner*

yang ditemukan pada sebuah logo maka tingkat akurasinya naik, demikian juga sebaliknya semakin sedikit *corner* yang ditemukan tingkat akurasinya menurun.

5. KESIMPULAN DAN SARAN

5.1. Kesimpulan

Dari hasil eksperimen dan evaluasi penelitian terhadap logo *recognition* diperoleh kesimpulan sebagai berikut:

a. Metode *FAST, SURF, dan FLANN TREE* akurat untuk melakukan identifikasi logo pada proses *feature recognition* berbasis *corner detection* yang terdapat di dalam sistem *augmented reality.*
b. Metode *FAST, SURF, dan FLANN TREE* memiliki akurasi rata-rata 73 % untuk kondisi normal, 69% untuk kondisi *scale invariant*, 65% untuk kondisi *rotation* 90^0 dan 66% untuk kondisi *rotation* 180^0.
c. Akurasi metode FAST, SURF, dan FLANN TREE memiliki tingkat rata – rata akurasi 14% lebih tinggi dibandingkan metode Fast, Brief dan Flann Tree dan 31% lebih tinggi dibandingkan metode Fast, Sift dan Flann Tree.

5.2. Saran

Berdasarkan hasil eksperimen dan evaluasi penelitian terhadap logo *recognition* maka diberikan beberapa saran sebagai berikut:

a. Metode *FAST, SURF, dan FLANN TREE* dapat digunakan untuk pengembangan sistem *augmented reality*.
b. Metode *FAST, SURF, dan FLANN TREE* yang akurat dapat diimplementasikan lebih lanjut pada pembuatan aplikasi *augmented reality* dalam perangkat *mobile*.

PENUTUP

Puji syukur penulis panjatkan kepada Tuhan Yesus Kristus yang telah *member*ikan anugrahNya dan kepada Roh Kudus yang telah *member*ikan inspirasi sehingga penulisan Jurnal yang berjudul "*feature RECOGNITION* BERBASIS *corner detection* dengan metode *FAST, SURF* DAN *FLANN TREE* UNTUK identifikasi logo pada *AUGMENTED REALITY MOBILE SYSTEM"* dapat terselesaikan dengan baik. Pada kesempatan ini penulis menyampaikan terima kasih yang tak terhingga kepada Dr. Ir. Edi Noersasongko, M.Kom selaku rektor Universitas Dian Nuswantoro dan segenap jajarannya.

PERNYATAAN ORIGINALITAS

"Saya menyatakan dan bertanggung jawab dengan sebenarnya bahwa Jurnal Ilmiah ini adalah hasil karya saya sendiri kecuali cuplikan dan ringkasan yang masing-masing telah saya jelaskan sumbernya". [Rastri Prathivi – P31.2010.00711]

DAFTAR PUSTAKA

[1] Mikolajczyk, K., Schmid, C. (2005). "Performance Evaluation of Local Descriptor," *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 27(10). pp. 1615-1630.
[2] Wei-Chao Chen, Yingen Xiong, Jiang Gao, Natasha Gelfand, Radek Grzeszczuk. (2007). "Efficient Extraction of Robust Image *Feature*s on Mobile Devices," *ISMAR*.
[3] Herbert Bay, Tinne Tuytelaars, Luc Van GooL. (2006). "Speeded-up Robust *Feature*," *Computer Vision and Image Understanding*. 110(3). 346-359.
[4] Gabriel Takacs, Vijay Chandrasekhar, Natasha Gelfand, Yingen Xiong, Wei-Chao Chen, Thanos Bismpigiannis, Thanos Bismpigiannis, Radek Grzeszczuk, Kari Pulli, dan Bernd Girod. (2008). "Outdoors Augmented Reality on Mobile Device Using Loxel-Based Visual *Feature* Organization," *Proc. of ACM international conference on multimedia information retrieval (ACM MIR).* Vancouver. Canada.

[5] Niels Henze, Torben Schinke, Susanne Boll. (2009). "What is That? Object Recognition from Natural *Feature*s on a Mobile Device," *Research Paper University of Oldenburg*.

[6] Daniel Wagner, Gerhard Reitmayr, Alessandro Mulloni, Tom Drummond, Dieter Schmalstieg. (2010). "Real Time Detection and Tracking for Augmented reality on Mobile *Device*," *IEEE Transaction on Visualization and Computer Graphic*. Vol 16. No 3.

[7] Charles Norona, Tyagi, Vivek Kumar. (2010). "Non-distributed Object Recognition Potential on the Android Platform," *COT5930 – Digital Image Processing Fall*.

[8] Mosalam Ebrahimi, Walterio W. Mayol-Cuevas. (2011). "Adaptive Sampling for *Feature* Detection, Tracking and Recognition on Mobile Platforms," *IEEE Transactions on Circuits and Systems for Video Technology*.

[9] Sami Mohammad Halawani, Dzulkifli Mohammad. (2011) "Logo *Matching* Technique Based on Principle Component Analysis," *International Journal of Computer Vision and Applications.* Vol 1. No 1. April 2011.

[10] Tom Drummond, Edward Rosten. (2006). "Machine *Learning* for High-Speed Corner Detection," *European Conf. Computer Vision (ECCV'06)*.pp. 430-443.