Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.201851630
Vol. 5, No. 1, Maret 2018, hlm. 104-113 p-ISSN: 2355-7699
Akreditasi RISTEKDIKTI, No 51/E/KPT/2017 e-ISSN: 2528-6579

PENCARIAN PRODUK YANG MIRIP MELALUI *AUTOMATIC ONLINE ANNOTATION* DARI *WEB* DAN BERBASISKAN KONTEN DENGAN *COLOR HISTOGRAM BIN* DAN *SURF DESCRIPTOR*

Putra Pandu Adikara[1], Sigit Adinugroho[2], Yuita Arum Sari[3]

[1] Fakultas Ilmu Komputer, Universitas Brawijaya, Malang
[2] Fakultas Ilmu Komputer, Universitas Brawijaya, Malang
[3] Fakultas Ilmu Komputer, Universitas Brawijaya, Malang
Email: [1]adikara.putra@ub.ac.id, [2]sigit.adinu@ub.ac.id, [3]yuita@ub.ac.id

(Naskah masuk: 23 Januari 2018, diterima untuk diterbitkan: 28 Februari 2018)

Abstrak

Banyaknya situs *e-commerce* memberikan kemudahan bagi pengguna yang ingin mencari dan membeli suatu produk, misalnya membeli makanan, obat, alat elektronik, kebutuhan sehari-hari, dan lain-lain. Pencarian suatu produk terhadap beberapa situs *e-commerce* akan menjadi sulit karena banyaknya pilihan situs, banyaknya penjual (merchant/seller) yang menjual barang yang sama, dan waktu yang lama karena harus berpindah-pindah situs hingga menemukan produk yang diinginkan. Selain itu dengan adanya teknologi *smartphone* berkamera, *augmented reality*, *query* pencarian bisa jadi hanya berupa citra, namun pencarian produk dengan menggunakan citra pada umumnya tidak diakomodasi di situs *e-commerce*. Dalam penelitian ini dikembangkan sistem meta *search-engine* yang menggunakan *query* berupa citra dan berbasiskan konten untuk menggabungkan hasil pencarian dari beberapa situs *e-commerce*. Citra *query* yang tidak diketahui namanya dibangkitkan *tag* atau kata kuncinya melalui Google *reverse image search engine.* Kata kunci ini kemudian diberikan ke masing-masing situs *e-commerce* untuk dilakukan pencarian. Fitur yang digunakan dalam pencocokan *query* dengan produk adalah fitur tekstual, *color histogram bin*, dan keberadaan citra objek yang dicari menggunakan SURF *descriptor*. Fitur-fitur ini digunakan untuk menentukan relevansi terhadap hasil penelusuran. Sistem ini dapat memberikan hasil yang baik dengan *precision@20* dan *recall* hingga 1 dengan rata-rata *precision@20* dan *recall* masing-masing sebesar 0,564 dan 0,608, namun juga bisa gagal dengan *precision@20* dan *recall* sebesar 0. Hasil yang kurang baik ini dikarenakan *tag* yang dibangkitkan terlalu umum dan situs *e-commerce*-pun memberikan hasil yang umum juga.

Kata kunci: *pencarian produk, penandaan citra, color histogram bin, SURF, reverse image search engine*

FINDING SIMILAR PRODUCT USING AUTOMATIC ONLINE ANNOTATION FROM WEB AND CONTENT-BASED USING COLOR HISTOGRAM BIN AND SURF DESCRIPTOR

Abstract

The vast number of e-commerce sites in Indonesia provides an easy solution for a consumer to buy a product, for example foods, medicine, electronic devices, etc. Searching a product in an e-commerce portal may be difficult due to large number of sites and sellers who sell similar products. Much time is spent to find matching product. In the other hand, with the latest development of camera phone and augmented reality, there is a possibility to use an image as a query although it is not yet accomodated by all e-commerce sites. This research develops meta search engine system using an image as an input and merge results from popular e-commerce sites. Tags or keywords are generated by Google reverse image search engine from a query image of unknown images. The keywords are then passed to e-commerce sites to get corresponding results. Features used in matching the query and the products are textural features, color histogram bin, and SURF descriptor to determine the existence of the query object. These features are used to determine the relevance of the search result. The system is able to produce good result, indicated by precision@20 and recall@20 value of 1 with average of precision@20 and recall value 0.564 and 0,608 respectively. However, the system can also fail with the value of precision@20 and recall@20 values are 0. The unsatisfied result occurs if the generated tags are too common and the e-commerce sites return common result.

Keywords: *product searching, image tagging, color histogram bin, SURF, reverse image search engine*

Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.201851630
Vol. 5, No. 1, Maret 2018, hlm. 104-113 p-ISSN: 2355-7699
Akreditasi RISTEKDIKTI, No 51/E/KPT/2017 e-ISSN: 2528-6579

1. PENDAHULUAN

E-commerce menjadi solusi untuk berbelanja online dan perkembangannya begitu pesat di Asia Pasifik, terutama di Indonesia. Bahkan di Asia, Indonesia memiliki pasar *e-commerce*, baik dalam bentuk *e-commerce Business-to-Customer* (B2C) begitu juga *Customer-to-Customer (C2C)*(Harsono, 2016). Banyak situs *E- Commerce* yang saling berkompetisi untuk mampu mengungguli satu sama lain dengan memberikan kelebihan-kelebihan bagi calon pembeli. Beberapa situs *e-commerce B2C* besar di Indonesia antara lain Bhinneka, Lazada Indonesia, sedangkan situs *C2C* antara lain Kaskus, TokoBagus, Berniaga, Tokopedia, serta situs jejaring sosial Facebook. Bagi calon pembeli sendiri dengan banyaknya situs *e-commerce* dapat membantu dengan memberikan berbagai pilihan dengan produk beragam serta dapat memesan di mana dan kapan saja tanpa repot harus datang ke tempat penjual.

Banyaknya situs yang tersedia menjadi kompleksitas tersendiri. Berikut ini adalah dua ilustrasi contoh kasus: (1) Dengan banyaknya pilihan situs *e-commerce* calon pembeli harus membuka satu per satu situs *e-commerce* untuk mencari produk yang dicari atau bila di beberapa situs tersedia maka calon pembeli harus membuka banyak situs dan membandingkan produk yang sama atau mencari yang mirip. (2) Apabila calon pembeli melihat suatu produk yang dijual di suatu tempat atau toko, berbekal citra produk yang di-*capture* (misalnya menggunakan teknologi *augmented reality smart glasses* seperti Google Glass terdahulu), calon pembeli ingin mengetahui produk yang serupa (dan mungkin tidak diketahui namanya) yang dijual di tempat lain. Dalam kedua kasus ini diasumsikan memiliki tujuan yang sama yaitu mengetahui harga yang paling murah dan kecocokan yang sesuai.

Masalah dalam kasus ilustrasi pertama dapat diatasi dengan memanfaatkan teknik seperti *meta search-engine* pada *information retrieval*. *Meta search engine* merupakan teknik menggabungkan hasil pencarian berdasarkan suatu *query* dari beberapa *search engine* dan menghasilkan satu hasil pencarian dengan urutan yang baru. Di sini *search engine* yang digunakan adalah *search engine* pada masing-masing *e-commerce*. Sayangnya untuk kasus ilustrasi kedua, masalah ini tidak dapat dipecahkan karena *search engine e-commerce* umumnya hanya mendukung pencarian dengan *query* berupa *keyword* bukan citra produk. Apabila nama produk tersebut tidak diketahui, maka terlebih dahulu diperlukan teknik untuk membuat *keyword* dari citra produk. Setelah diketahui *keyword* dari citra produk baru diberikan ke *search engine* tiap *e-commerce* dan hasil pencariannya digabungkan. Namun, hanya mengandalkan *keyword* saja dapat menyebabkan hasil pencarian memberikan produk yang berbeda secara visual dari yang diinginkan atau sulit untuk menemukan produk yang berbeda namun mirip secara visual.

Dari penelitian berbasiskan konten untuk mencari produk yang mirip secara visual dalam hal ini adalah pakaian, pencarian umumnya dilakukan secara *offline* menggunakan basis data sendiri (*in-shop*) bukan secara *online* dan tidak menggabungkan beberapa hasil pencarian dari beberapa *online shop* (Hsu, Paz dan Shen, 2001). Penelitian lain yang menggunakan pencarian berbasiskan konten juga telah dilakukan (Liu et al., 2016; Kiapour et al., 2015). Dari penelitian-penelitian yang telah dilakukan, umumnya pencarian produk berupa produk fashion seperti pakaian, aksesoris namun tidak untuk produk-produk yang lain yang lebih umum. *Annotation* atau pemberian label dan kategori yang digunakan dalam data latih ataupun data uji pun masih dilakukan secara manual oleh manusia menggunakan *bounding box* dari produk. Masih belum ada pemberian *annotation* yang dapat dilakukan secara otomatis dengan memanfaatkan sumber daya yang ada yaitu dari *web*. Oleh karena itu dalam penelitian ini akan berfokus pada *meta search–engine* dengan *query* berupa citra produk dan pencocokan secara *content-based* untuk mengatasi masalah yang diungkapkan di atas. Namun, karena dilakukan secara *online* maka diperlukan teknik pencocokan berbasiskan konten yang cepat pula. Dalam penelitian ini akan digunakan fitur warna serta fitur lokal yang stabil *(local invariant feature)* yang cepat untuk didapatkan dan dicocokkan. Fitur warna digunakan untuk mencocokkan kemiripan visual berdasarkan warna. Namun, perhitungan ukuran kemiripan/jarak dari fitur warna dapat memengaruhi hasil kemiripan dua citra sehingga ukuran ini merupakan salah satu faktor penting. Kemiripan warna pada fitur citra dihitung berdasarkan histogram menggunakan ukuran *distance Chi-square* yang merupakan ukuran *distance* yang lebih baik setelah *Earth Moving Distance* (EMD). EMD meskipun memberikan hasil yang lebih baik dari *Chi-square* (Rubner, Tomasi dan Guibas, 2000) tetapi membutuhkan waktu yang lama dibandingkan *distance* lain sehingga tidak cocok untuk sistem yang memerlukan waktu komputasi yang cepat.

Local invariant feature digunakan untuk mencocokkan objek yang dicari berdasarkan fitur yang stabil dan tidak mudah berubah karena perubahan objek pada citra, misalnya pencahayaan, rotasi, skala, dll. Dari beberapa penelitian *Scale-Invariant Feature Transform (*SIFT*)* merupakan *local invariant feature* yang baik namun karena memerlukan waktu komputasi yang banyak maka SIFT tidak memenuhi syarat untuk aplikasi *real time* (Juan dan Gwun, 2009), (Pena, 2012). Oleh karena itu, algoritme untuk mendapatkan *local invariant feature* yang digunakan adalah yaitu *Speeded Up Robust Features (*SURF*)* karena memiliki kelebihan antara lain pencocokan yang cepat beberapa kali dari SIFT (Juan dan Gwun, 2009)dan invarian terhadap rotasi, skala dan lebih tahan terhadap kekaburan (*blur)* dan perubahan pencahayaan (*luminance*).

Berdasarkan permasalahan yang telah diuraikan, pada penelitian ini diusulkan sistem temu kembali citra untuk pencarian produk melalui *automatic online*

annotation dari web dan berbasiskan konten dengan *color histogram bin* dan SURF *descriptor*. Diharapkan dengan rancangan sistem dan metode yang diusulkan, maka pencarian produk yang tidak diketahui namanya dapat dilakukan dan akan menghasilkan produk-produk yang mirip atau bahkan sama sesuai relevansinya pada *e-commerce*.

2. TINJAUAN PUSTAKA

2.1. Pembangkitan *Annotation/Tag* Citra

Citra memiliki *tag* atau kata-kata yang berasosiasi dengan citra tersebut. *Tag* atau kata dapat diekstrak dengan cara menerjemahkan citra berdasarkan objek-objek yang terkandung di dalamnya (Hou et al., 2010). Dari hasil segmentasi objek, fitur visualnya kemudian menjadi *input* ke *classifier* sehingga menghasilkan *output* anotasi otomasi. *Search engine*, misal Google, mampu membangkitkan *tag* dari sebuah gambar masukan. Pemanfaatan *search engine* ini dilakukan untuk menghindari *reinventing the wheel*, daripada membuat indeks basis data citra yang sangat besar untuk berbagai objek atau produk. *Keyword* ini nantinya juga dijadikan sebagai fitur tekstual yang dibandingkan dengan hasil pencarian produk.

2.2. Kemiripan Citra

Kemiripan citra dilakukan untuk mengetahui apakah suatu citra memiliki kemiripan dengan citra lainnya. Citra pertama dengan citra kedua bisa jadi memiliki perbedaan perspektif karena adanya skala, rotasi, cara pandang, atau bisa jadi citra pertama merupakan patch atau bagian dari citra kedua. Kemiripan citra ini menjadi sangat penting dan potensial untuk berbagai aplikasi, misalnya *content based information retrieval* (CBIR), *optical character recognition* (OCR), robotika, analisis citra medis, pengenalan biometrik seperti sidik jari, iris mata, wajah, dll. Kemiripan citra dapat dihitung melalui fitur-fitur yang diekstraksi dari citra. Dalam penelitian ini digunakan fitur warna serta fitur *local invariant*. Fitur lain seperti bentuk atau tekstur tidak digunakan dalam penelitian ini karena diasumsikan membutuh-kan waktu yang lama untuk perhitungan padahal aplikasi yang dibutuhkan memiliki syarat perhitungan yang cepat. Dalam penelitian ini kemiripan citra akan dilakukan dengan menggunakan pencocokan fitur warna dan pencocokan objek.

2.3. Pencocokan Fitur Warna

Warna merupakan fitur visual yang paling banyak digunakan. Penggunaan histogram warna merupakan cara yang umum untuk merepresentasikan fitur warna. Histogram warna adalah distribusi dari banyak *pixel* pada citra. Banyak elemen pada histogram tergantung dari banyak bit dari tiap *pixel* citra. Misalnya *pixel* dengan kedalaman *n* bit, maka nilai *pixel* antara 0-2^n-1 dan histogram memiliki 2^n elemen.

Ruang warna dalam penelitian ini menggunakan ruang warna *Hue, Saturation, Value* (HSV) atau dikenal juga *Hue, Saturation, Lightness* (HSL). Ruang warna HSV ini lebih natural terhadap persepsi mata manusia dibandingkan ruang warna *Red, Green, Blue* (RGB) (Sural, Qian, Pramanik dan Lansing, 2002). HSV memisahkan tingkat kecerahan pada *channel* yang berbeda, yaitu *channel* Value. Dengan pemisahan ini maka suatu citra nantinya dapat dianggap mirip meskipun berbeda tingkat kecerahannya.

Penelitian (Rubner, Tomasi dan Guibas, 2000) menunjukkan bahwa perhitungan *similarity* terhadap fitur warna berupa *color histogram* menentukan hasil kemiripan citra. Oleh karena itu ukuran *similarity* juga menjadi faktor penting untuk memberikan hasil kemiripan berdasarkan warna. Salah satu metode pencocokan fitur warna adalah statistik *Chi Square* X^2. *Chi-square* digunakan untuk mengukur seberapa beda suatu distribusi yang diambil dari populasi diwakili oleh yang lain (yang diamati dengan yang diharapkan). Kemiripan antara dua histogram, *H* dan *K*, dapat ditentukan dengan metode Chi-square pada Persamaan (1).

$$d_{x^2}(H,K) = \sum_i \frac{(h_i - m_i)^2}{m_i},\ m_i = \frac{h_i + k_i}{2} \quad (1)$$

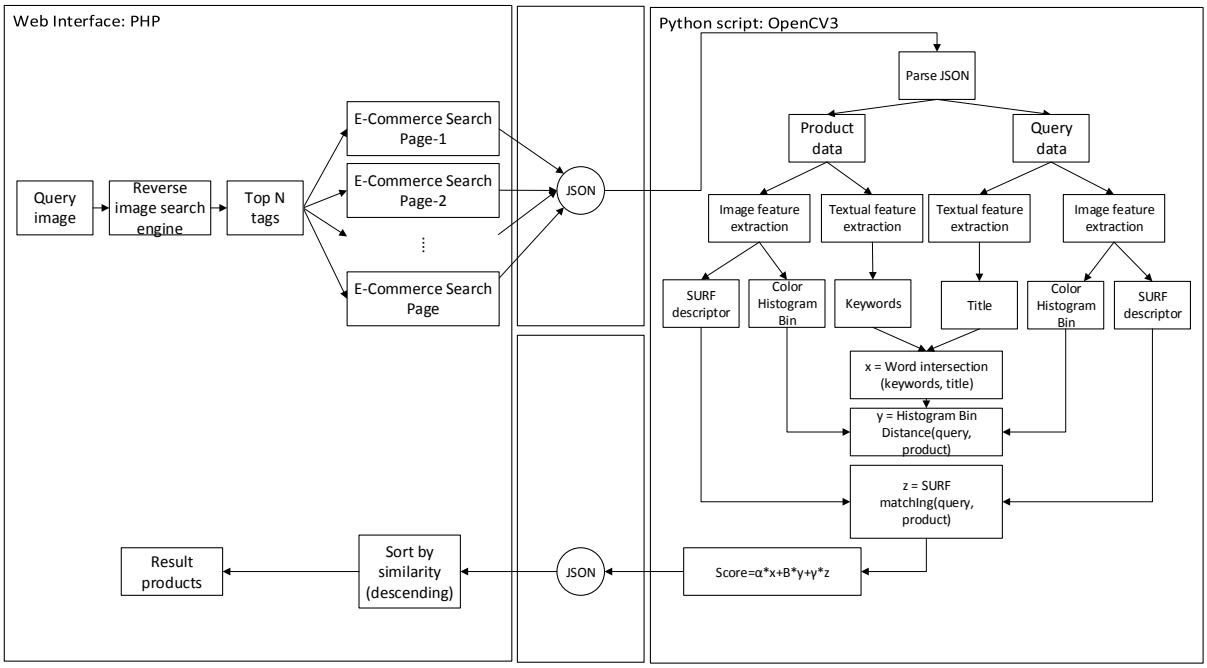
Keterangan:
H = histogram pertama
K = histogram kedua
h_i = *color bin* pada histogram pertama
k_i = *color bin* pada histogram kedua

2.4. *Speeded Up Robust Feature* (SURF)

Pencocokan objek pada citra dalam penelitian ini menggunakan algoritme *Speeded Up Robust Features* (SURF). *Speeded Up Robust Features* (SURF) adalah algoritme yang cepat dan handal untuk perbandingan dan representasi gambar yang mirip melalui *detector* dan *descriptor* yang invarian terhadap skala dan rotasi (Bay, Ess, Tuytelaars dan Van Gool, 2008). Algoritme ini berdasarkan tesis H. Bay yang merupakan "perbaikan" dari algoritme sebelumnya, SIFT. SURF memiliki kemiripan dengan pendekatan SIFT yang memilih *interest point* pada citra dari fitur-fitur yang menonjol dalam ruang skala (*scale space*) linear kemudian membuat fitur-fitur lokal berdasarkan distribusi gradien. SURF memiliki kelebihan karena komputasinya yang cepat dari aproksimasi operator diferensial dalam ruang skala, sehingga memungkinkan digunakan dalam aplikasi *real time*. Algoritme SURF memiliki tiga langkah dalam pencocokan citra:

1. Mendeteksi *interest point*

Deteksi fitur pada SURF mengandalkan *integral images* untuk mengurangi waktu komputasi dan disebut *Fast-Hessian Detector*. Supaya dapat mendeteksi *feature point* yang invarian terhadap skala maka digunakan pendekatan *cascading-filter* yang mana *Difference of Gaussian* (DoG) dihitung secara progresif terhadap citra yang di-*downscale*. Teknik yang memungkinkan invarian terhadap skala adalah memeriksa citra pada beberapa skala yang berbeda (*scale-space*) menggunakan kernel Gaussian.



Gambar 1 Arsitektur dari sistem yang dikembangkan

SURF menggunakan aproksimasi *scale space* yang disebut sebagai *box space* dan untuk mempercepat komputasi konvolusi dilakukan terhadap citra integral. SURF membagi *scale space* ke beberapa *level* dan *octave*. *Octave* merepresentasikan serangkaian *response map* yang diperoleh dari konvolusi citra *input* dengan filter yang membesar ukurannya. Satu *octave* merupakan faktor skala 2 (penggandaan) dari σ dan tiap *octave* dibagi ke beberapa *level* skala, misal dalam 1 *octave* ada 3 *level* σ=k, σ=3/2k σ=2k. Dari citra pada beberapa *level* dalam *octave* dilakukan operasi konvolusi sehingga menghasilkan piramida *response map* diawali dengan filter 9x9 untuk σ=1.2 kemudian ukuran filter naik 6 atau lebih dengan menjaga struktur filter.

Interest point adalah titik-titik ekstrema dalam 8 tetangga dalam tingkat yang sama dan 2x9 tetangga pada tingkat di atas dan di bawah. SURF menggunakan *blob detector* berdasarkan Hessian yang ditunjukkan pada Persamaan (2), Persamaan (3), Persamaan (4) untuk menentukan *interest point*. Determinan dari matriks Hessian menunjukkan perubahan lokal di sekitar area.

$$\mathcal{H}(x,\sigma) = \begin{bmatrix} L_{xx}(x,\sigma) & L_{xy}(x,\sigma) \\ L_{xy}(x,\sigma) & L_{yy}(x,\sigma) \end{bmatrix} \quad (2)$$

$$L_{xx}(x,\sigma) = I(x) * \frac{\partial^2}{\partial x^2} g(\sigma) \quad (3)$$

$$L_{xy}(x,\sigma) = I(x) * \frac{\partial^2}{\partial xy} g(\sigma) \quad (4)$$

Keterangan:

$L_{xx}(x, \sigma)$ = konvolusi dari citra dengan turunan order kedua Gaussian $\frac{\partial^2}{\partial x^2} g(\sigma)$ pada citra *I* di titik *x*.

x = titik pada citra (yang memiliki koordinat *x, y*)

σ = skala dari citra yang dikonvolusi.

2. Membuat *descriptor* dari tiap interest point.

Descriptor bertujuan untuk menyediakan deskripsi unik dan handal terhadap suatu fitur dan dapat dibuat berdasarkan area sekitar *interest point*. Pada SURF, *descriptor* dibuat berdasarkan Haar wavelet. Pentingnya menentukan orientasi yang unik dari *interest point* supaya dapat invarian terhadap rotasi. Untuk dapat mengatasi invarian terhadap rotasi, orientasi dominan didefinisikan dengan distribusi gradien lokal yang dinilai dengan Haar wavelet.

Descriptor SURF menjelaskan area *interest* dengan ukuran 20s yang dibagi menjadi 4x4 sub-area. Sub-area ini dideskripsikan dengan nilai dari respons *wavelet* terhadap arah x dan y. Respons *wavelet* terhadap *x* adalah *dx* dan terhadap *y* adalah *dy*. Untuk tiap subarea vektor $v = \{\sum dx, \sum|dx|, \sum dy, \sum|dy|\}$ dilakukan perhitungan berdasarkan sample 5x5. *Descriptor* dari *interest point* berdimensi 16x4 yang mana 16 vektor merupakan penggabungan beberapa sub-area. *Descriptor* kemudian dinormalisasi supaya invarian terhadap variasi kontras. Untuk pencocokan SURF ditambahkan *sign* Laplacian.

$$\nabla^2 L = L_{xx}(x,\sigma) + L_{yy}(x,\sigma) \quad (5)$$

Laplacian menghasilkan respons positif untuk *blob* gelap dan respons negatif untuk *blob* terang. Hal ini digunakan untuk membedakan antara *blob* terang di latar belakang gelap.

3. METODOLOGI PENELITIAN

3.1. Arsitektur Sistem

Arsitektur sistem dapat dilihat pada Gambar 1. Sistem yang dikembangkan sebenarnya adalah sistem temu kembali citra berbasiskan *web* dan bersifat *online*. Sistem ini harus *online* karena pembuatan

query menggunakan *annotation/tag* dari citra memerlukan *reverse image search engine* serta untuk pencarian produk yang ada di situs *e-commerce*. *Reverse image search engine*, misalnya Google Images memiliki *dataset* yang sangat besar sehingga dapat dipergunakan atau dieksploitasi daripada harus membuat *dataset* baru dan *crawling* seluruh gambar yang ada di Internet. Citra produk *Q* yang tidak diketahui namanya dibangkitkan *tag*-nya oleh *reverse image search engine*, diekstraksi fitur citra, lalu dicocokkan dan ditampilkan hasil temu kembalinya. Sistem ini diimplementasikan dalam *script* Python untuk *back-end* dan *front-end web* menggunakan PHP.

3.2. Pembangkitan *Tags/Keywords*

Pembangkitan *tag* atau *keyword* dilakukan pada langkah awal dalam pencarian (Pseudocode 1). Pembangkitan dilakukan melalui Google Images yang mendukung pencarian berdasarkan *query* berupa citra dan diasumsikan memiliki basis data citra yang sangat besar. Google Images memberikan hasil pencarian berupa situs yang berisi gambar yang sama atau mirip, *best guess* dari suatu citra (apabila tersedia), serta citra-citra lain yang mirip secara visual (apabila tersedia). Halaman hasil pencarian dari Google Images kemudian disimpan dalam bentuk file HTML. Halaman ini kemudian akan di-*parse* menggunakan XPath. XPath adalah bahasa *query* untuk navigasi atau memilih suatu node dari halaman XML. HTML dapat juga dianggap sebagai dokumen XML sehingga XPath juga dapat diterapkan pada file HTML. XPath ini digunakan untuk mendapatkan beberapa informasi berikut:

1. *Best guess* (bila tersedia).
2. Judul, URL, dan cuplikan dari situs yang muncul di hasil pencarian.

Pseudocode 1. Fungsi untuk mendapatkan *tags* dari suatu citra

```
1  function GET_TAGS(query_image):
2    ▷ menghasilkan suggested tags, bila
   MAX_N_WORDS = 4, MIN_N_WORDS = 2
3      suggested_tags[0] = "kata1 kata2 kata3 kata4"
4      suggested_tags[1] = "kata1 kata2 kata3"
5      suggested_tags[2] = "kata1 kata2"
6    page = simpan halaman pertama dari Google
           reverse image search engine dari query_image
7    best_guess[] = tokenisasi kata-kata pada Google
   best guess dari page
8    words[] = tokenisasi semua kata pada page
9    for w in words do
10     hitung tf-idf dari w        ▷ 1 link hasil
           pencarian dianggap sebagai 1 dokumen
11   end for
12   i = 0
13   for j=MAX_N_WORDS to MIN_N_WORDS do
14     top_n[] = ambil kata dari words, yang telah
           diurutkan menurun sebanyak j kata
15     tags = intersection kata dari best_guess dan
           top_n (cari yang overlap)
16     tags = gabungkan array kata top_n dan tags
17     suggested_tags[i] = tags
18     i = i+1
19   end for
20   return suggested_tags
21 end function
```

3. Pencarian produk pada *search engine e-commerce.*

Pencarian kemudian dilakukan ke *search engine* beberapa *e-commerce* berdasarkan *keyword* yang dibangkitkan. Dalam penelitian ini ditentukan beberapa *e-commerce* besar di Indonesia. Pada masing-masing situs tersebut dilakukan pencarian berdasarkan metode HTTP GET.

Berikut ini daftar situs *e-commerce* dan *query* GET yang didukung:

1. Lazada Indonesia (http://www.lazada.co.id/catalog/?q=***keyword)***
2. Bhinneka (http://www.bhinneka.com/search.aspx?Search=***keyword)***
3. BukaLapak (http://www.bukalapak.com/products?utf8=%E2%9C%93&search[keywords]=***keyword)***
4. Tokopedia (http://www.tokopedia.com/search?q=***keyword***)

Halaman hasil pencarian berdasarkan *keyword* pada masing-masing situs *e-commerce* kemudian disimpan dalam bentuk HTML dan di-*parse* menggunakan XPath. Beberapa metadata produk yang didapat dari hasil *parsing* antara lain:

1. Nama produk
2. URL dari produk
3. URL citra produk
4. Harga
5. ID/Nama *e-commerce*

Setelah didapatkan URL citra produk, *file* citra kemudian disimpan juga. Keseluruhan informasi *keyword*, citra *query* Q dan metadata produk serta citra produk kemudian disimpan dalam file *result.json*. File *result.json* ini yang antinya akan di-parsing oleh *script* Python untuk mendapatkan fitur-fitur berbasiskan konten citra.

3.3. Ekstraksi Fitur Citra

Script Python kemudian melakukan *parsing* terhadap *result.json* untuk memproses fitur citra dengan bantuan modul OpenCV3.

1. Fitur warna

Fitur warna berupa *color histogram bin* didapatkan dari citra produk dan citra *query* menggunakan Pseudocode 2. Awalnya citra di-*downscale* ke ukuran 64x64 piksel lalu diubah *color space*-nya ke HSV. *Color space* yang digunakan pada penelitian ini adalah *Hue, Saturation, Value* (HSV) atau *Hue, Saturation, Lightness* (HSL). Penggunan HSV daripada *Red, Green Blue* (RGB) karena RGB rentan terhadap pencahayaan (*illumination*), sedangkan pada HSV, *channel* untuk pencahayaan dipisahkan tersendiri pada *channel Lightness/Value*. Setelah menjadi citra HSV kemudian buat histogram bin pada *channel hue* dan *saturation* dan lakukan normalisasi. Pada OpenCV perhitungan *histogram bin count* dibantu dengan fungsi `calcHist`.

Pseudocode 2. Fungsi untuk mendapatkan fitur *color histogram bin*

```
1  function GET_COLOR_HISTOGRAM_BIN(image):
2      image = resize image ke 64x64
3      image = konversi color space dari image ke HSV
4      buat histogram bin untuk channel hue dari image,
           dengan bin = 16
5      buat histogram bin untuk saturation channel dari
           image, dengan bin = 20
6      lakukan normalisasi histogram [0,1] dari channel
           hue dan channel saturation
7      return bin dari channel hue dan channel
           saturation
8  end function
```

2. Fitur *local invariant* SURF

Untuk mendapatkan fitur SURF *descriptor* pada OpenCV digunakan fungsi yang telah disediakan yaitu `surf = cv2.xfeatures2d.SURF_create()` dilanjutkan `surf.detectAndCompute(img, None)`. Untuk mendeteksi SURF *point*, citra sebelumnya diubah menjadi *grayscale*. Setelah didapatkan SURF *point* baru dilakukan ekstraksi fitur.

3.4. Perhitungan Nilai Kemiripan Fitur Citra

Hasil perhitungan nilai kemiripan fitur citra warna dan SURF *descriptor* ini nantinya akan disimpan dalam *distance.json* beserta informasi yang sama pada *result.json*.

1. Fitur tekstual

Fitur tekstual didapatkan dari judul produk yang kemudian dihitung nilai kemiripannya berdasarkan jumlah kata yang *overlap* dengan *keyword* (WI). Semakin banyak jumlah kata yang sama maka dianggap lebih relevan daripada jumlah kata yang lebih sedikit. Fungsi ini ditunjukkan pada Pseudocode 3.

Pseudocode 3. Fungsi untuk menghitung kemiripan berdasarkan fitur tekstual

```
1  function GET_TEXTUAL_FEATURE(query_tags,
       product_title):
2      n = hitung banyaknya kata yang overlap
           (intersection) dari tags hasil pembangkitan
           dan title pada product
3      return n
4  end function
```

2. Fitur warna

Perhitungan *distance* antara dua *color histogram bin* antara citra *query* dan produk dihitung menggunakan rumus Chi-Square. Fungsi perhitungan kemiripan ini ditunjukkan pada Pseudocode 4.

Pseudocode 4. Fungsi untuk menghitung kemiripan berdasarkan *distance* pada *color histogram bin*

```
1  function
       GET_COLOR_HISTOGRAM_BIN_FEATURE(
       img1, img2):
2      bin_img1 =
       GET_COLOR_HISTOGRAM_BIN(img1)
3      bin_img2 =
       GET_COLOR_HISTOGRAM_BIN(img2)
4      return DISTANCE_CHI-SQUARE(bin_img1, bin_img2)
5  end function
```

3. Fitur *local invariant* SURF

Perhitungan kecocokan SURF *descriptor* antara citra *query* Q dan citra produk P dilakukan menggunakan `flann = cv2.FlannBasedMatcher()` dengan fungsi `flann.knnMatch(descriptor1, descriptor2)`. Setelah dilakukan pencocokan SURF *descriptor*, maka selanjutnya dilakukan estimasi transformasi geometrik dari pasangan SURF *point* yang cocok dengan menghilangkan outlier melalui fungsi `cv2.findHomography`. Tranformasi tersebut dilakukan untuk melokalisasi objek yang dicari. *"Bounding polygon"* kemudian dibentuk melalui bantuan `cv2.perspectiveTransform()`. Hasil transformasi *bounding polygon* mengindikasikan objek yang dianggap cocok antara citra *query* Q dan citra produk P. Apabila *match point* tidak memenuhi syarat terbentuknya *bounding polygon* (dengan luas area *polygon* > 2500) maka objek dianggap tidak ada. Nilai kemiripan SURF akan diberikan 1 apabila ada objek yang dianggap cocok dan 0 apabila tidak ada yang ditunjukkan pada Pseudocode 5.

Pseudocode 5. Fungsi untuk menghitung kemiripan adanya objek dengan SURF

```
1  function GET_SURF_FEATURE(query_image,
       product_image):
2      MIN_MATCH_COUNT = 10 //threshold untuk
       minimal kecocokan point SURF
3      found = 0          ▷ nilai default bila tidak
       ditemukan
4      des1 = fitur SURF descriptor dari query_image
5      des2 = fitur SURF descriptor dari product_image
6      good = cari kecocokan dengan
         FlannBasedMatcher melalui knnMatch(des1,
         des2) dan simpan hanya point yang baik
         berdasarkan Lowe's ratio test
7      if LENGTH(good) > MIN_MATCH_COUNT
       then
8          cari homography dari hasil point keypoint pada
       query_image dan product_image
9          if ditemukan homography then
10             hitung area dari polygon homography
11             ▷ menghindari false positive
               if luas area polygon > 3000 then
12                 ▷ bila image query ditemukan pada image
                   produk
                   found = 1
13             end if
14         end if
15     end if
16     return found
17 end function
```

4. Kombinasi fitur citra

Hasil akhir kemiripan didapat melalui penjumlahan fitur tekstual, ftur warna, dan fitur SURF. Penghitungan ketiga fitur ini nantinya akan dijumlahkan untuk mendapat skor total dikalikan dengan suatu konstanta untuk masing-masing nilai fitur seperti yang ditunjukkan pada Persamaan (6).

$$product_{total\_score,i} = \alpha * product_{text,\,i} + \beta * product_{surf,\,i} + \gamma * (\max(color) - product_{color,i}) \quad (6)$$

5. Penyajian Hasil Pencarian

Hasil *distance.json* yang berisi informasi *keyword*, *query*, produk beserta nilai kemiripannya akan disajikan kembali melalui antarmuka *web*. Penyajian hasil pencarian ini akan diurutkan berdasarkan nilai kemiripan. Dalam implementasinya ini, penyajian akan diurutkan secara *descending*. Fungsi ini ditunjukkan pada Pseudocode 6.

Pseudocode 6. Fungsi untuk melakukan pencarian dengan fitur-fitur yang diusulkan

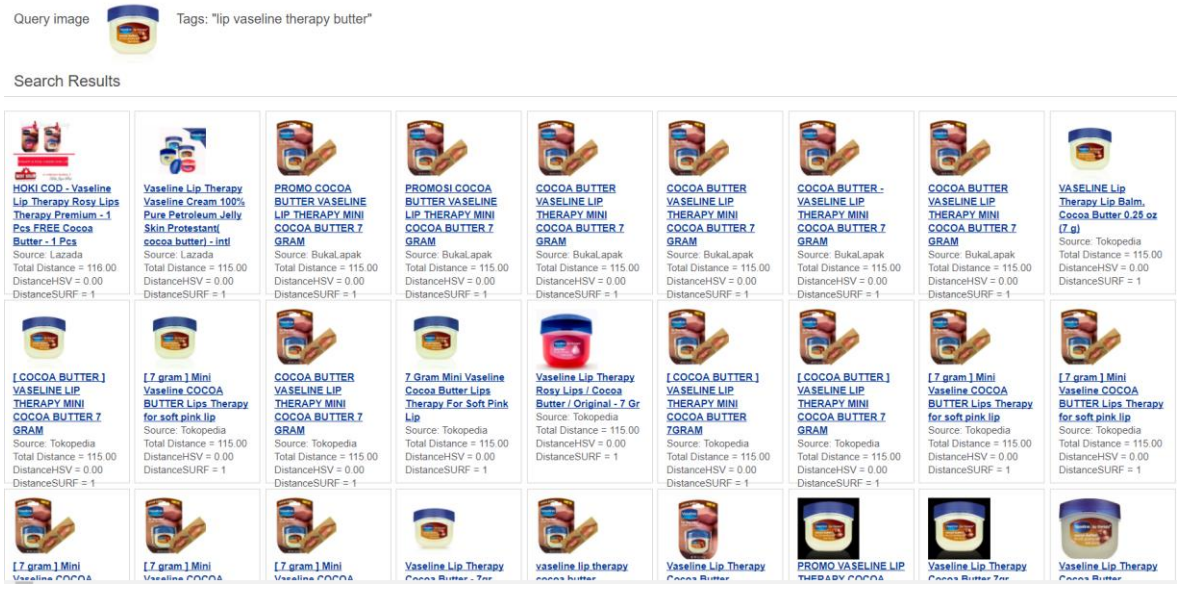
```
1  procedure SEARCH:
2    suggested_tags[] = GET_TAGS(query_image)
3    do
4      ▷ bila tidak ditemukan hasil pencarian
   (atau < 10) menggunakan suggested_tag[0], maka
   gunakan suggested_tag[1], dst
5      page = halaman pertama hasil search engine
         tiap e-commerce dengan keywords dari
         suggested_tag[index]
6      products[] = parse tiap produk dari page
7      index = index + 1
8    while(LENGTH(products) < 10)
9    max_color = -1
10   for product in products do
11     alpha = 1
12     beta = 100
13     gamma = 10
14     product_text = GET_TEXTUAL_FEATURE
         (query_tags, product_title)
15     product_surf =
   GET_SURF_FEATURE(query_image, product_image)
16     product_color =
   GET_COLOR_HISTOGRAM_BIN_FEATURE(
   query_image, product_image)
17     if product_color > max_color then
18       max_color = product_color
19   end for
20   for product in products do
21     product_total_score = alpha*product_text +
         beta*product_surf + gamma*(max_color-
         product_color)
22   end for
23   products = urutkan products berdasarkan
         product_total score secara menurun
24   tampilkan products
25 end procedure
```

4. HASIL PENGUJIAN

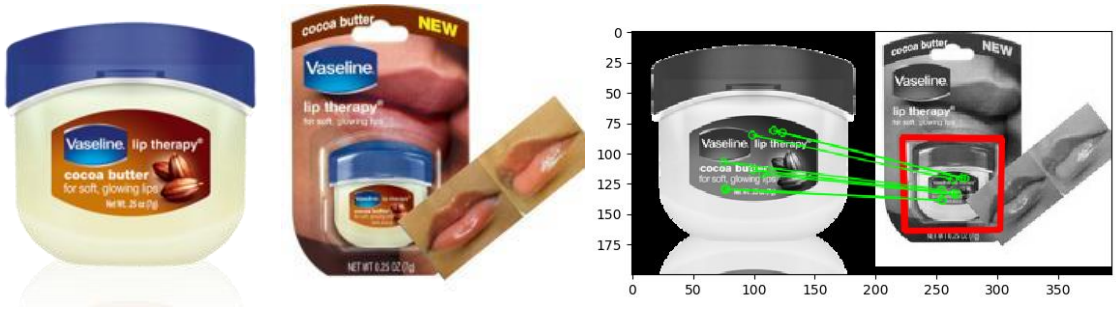
Eksperimen terhadap sistem *meta search engine* dengan *content based* ini dilakukan beberapa kali percobaan. Citra *query* yang digunakan adalah citra produk yang didapat secara *random* dengan resolusi yang bervariasi tergantung dari masing-masing *e-commerce*. Citra produk yang diambil ini muncul dalam produk rekomendasi di beberapa *e-commerce* antara lain Bhinneka, BukaLapak, Lazada, dan Tokopedia pada bulan Oktober 2017. Eksperimen pencarian dilakukan secara *real time* tanpa adanya data latih sehingga memberikan hasil agregasi yang bervariasi pula tergantung hasil pencarian yang diberikan masing-masing *e-commerce*.

Dalam eksperimen kecil ini digunakan 10 citra *query*. Citra yang digunakan diambil dari beberapa kategori antara lain gadget dan elektronik, peralatan rumah tangga, fashion, kesehatan dan kecantikan, olahraga dan musik. Contoh hasil pencarian yang dihasilkan oleh sistem tampak pada Gambar 1.

Dari contoh pada Gambar 1 didapatkan hasil produk yang sesuai dengan citra *query*, meskipun ada perbedaan pada citra. Contoh perbedaan citra antara *query* dan produk jelas terlihat pada Gambar 2. Pada Gambar 2 (b) terlihat di dalamnya terdapat objek pada citra Gambar 2 (a). Dengan menggunakan SURF, maka objek citra *query* dapat dideteksi pada citra produk meskipun memiliki perbedaan skala.



Gambar 1 Contoh hasil pencarian dari sistem yang dikembangkan.



(a) (b) (c)

Gambar 2 (a) citra *query*, (b) citra produk, (c) Hasil pencocokan berdasarkan SURF.

Pada Gambar 1 juga dapat terlihat bahwa ada citra produk yang memiliki *background* warna hitam, sedangkan citra *query* memiliki *background* warna putih. Dalam kasus ini fitur *color histogram bin* menggunakan *color space* HSV juga berperan penting untuk mengatasi perbedaan *lightness.* Hal ini juga terlihat pada citra produk paling kanan bawah yang redup dibanding citra *query* tetapi juga tetap dapat diperoleh oleh sistem yang dikembangkan.

Dari tiap eksperimen yang dilakukan kemudian dihitung nilai *precision* dan *recall*. Hasil *precision* dan *recall* eksperimen ditunjukkan pada **Error! Reference source not found.**. *Precision* dihitung menggunakan *Precision*@20, supaya memudahkan perhitungan evaluasi sistem dan membatasi bahwa yang dihitung hanya *top* 20 saja.

Dari hasil eksperimen dapat diketahui bahwa nilai *precision* dan *recall* sangat bervariasi, sehingga sulit diketahui akurasinya terhadap kategori tertentu. Hal ini juga dikarenakan citra produk yang bervariasi. Citra suatu produk bisa sangat mirip dan hanya berbeda sangat sedikit. Pada kasus yang terjadi pada nomor 4, produk yang dicari adalah "*popok*", produk-produk yang didapatkan menghasilkan citra yang sangat mirip, perbedaannya hanya terletak pada ukuran (S, M, L, XL) dan banyaknya isi saja. Dalam kasus ini perhitungan hanya dilakukan berdasarkan ukuran yang sama saja, sehingga hanya menghasilkan *precision@20* sebesar 0.45 dan *recall* sebesar 0.5. Apabila mengabaikan ukuran dan hanya berdasarkan citra saja, maka *precision@20* dan *recall*-nya pada 107 hasil pencarian memberikan nilai lebih tinggi yaitu 0.6 dan 1.

Citra produk yang terkenal atau populer cenderung mendapatkan *recall* yang tinggi (meski bisa jadi *precision*-nya rendah). Sedangkan citra produk yang mungkin tidak terkenal atau tidak dikenali Google Images maka menghasilkan *precision* dan *recall* yang rendah. Semua ini tergantung dari bagaimana Google menerapkan algoritme untuk melakukan *crawling* gambar dan memberikan hasil pencarian. Apabila Google tidak pernah mengindeks suatu citra beserta *webpage*-nya maka *online annotation* terhadap suatu gambar juga akan gagal. Begitu juga apabila meski pernah mengindeks, namun algoritme Google menempatkannya pada hasil relevansi yang rendah, maka tidak akan pernah muncul pada hasil pencarian pada halaman pertama. Padahal sistem yang dibangun ini membangkitkan *tag* dari citra berdasarkan hasil pencarian yang diberikan oleh Google pada halaman pertama.

Hal ini terlihat pada percobaan ke-9 dan dalam pencarian suatu *dress*/gaun berwarna pink, ternyata hasil yang diberikan oleh *e-commerce* tidak ada yang sesuai, sehingga *precision@20* dan *recall* sama-sama bernilai 0. Salah satu penyebabnya adalah pembangkitan *keywords* yang terlalu umum, hanya berupa "*dress gaun bahan*". Terlihat jelas bahwa hasil pembangkitan *keywords* sangat menentukan hasil pencarian. Semakin spesifik suatu *keywords* (misal dari merk hingga tipe produk), maka hasil yang didapatkan semakin relevan begitu juga sebaliknya. Namun apabila terlalu spesifik bisa jadi tidak ditemukan hasilnya karena *keywords* yang dibangkitkan ada yang tidak sesuai dari sebenarnya. Beberapa hasil eksperimen pencarian memberikan hasil kurang baik bahkan gagal berdasarkan sistem yang dikembangkan, namun ada pula hasil yang sangat baik. Hal tersebut dipengaruhi oleh pembangkitan kata kunci yang tepat supaya dapat memberikan hasil yang tepat pada *search engine e-commerce* dan ciri unik tekstur di dalam citra tampak signifikan atau tidak. Metode *keypoint* SURF tidak terlalu cocok untuk gambar yang cenderung memiliki tekstur polos atau warna yang homogen.

5. PENUTUP

Dari hasil penelitian telah dikembangkan sistem pencarian produk berdasarkan suatu citra yang mana citra tersebut tidak diketahui namanya. Sistem ini dapat memberikan hasil pencarian berdasarkan hasil agregasi hasil pencarian dari beberapa *e-commerce* berdasarkan tingkat relevansinya dari citra produk meski awalnya citra produk yang dicari tidak diketahui namanya. Sistem ini memanfaatkan *reverse image search engine* yang digunakan untuk membangkitkan *keywords/tags* dari suatu citra *query*. Dari *keywords/tags* inilah nantinya akan digunakan sebagai keywords untuk dilemparkan ke fitur pencarian suatu situs *e-commerce*.

Hasil pencarian masing-masing situs *e-commerce* akan diberikan dikumpulkan dan dilakukan pembobotan.

Tabel 1 Hasil *precision* dan *recall* dari eksperimen

No	Citra *query*	Kategori	*Precision @20*	*Recall*
1		Gadget	0,67	1
2		Kesehatan	0,45	0,5
3		Gadget	0,67	0,43
4		Kesehatan	1	1
5		Elektronik	1	1
6		Gadget	0,45	0,86
7		Kecantikan	0,7	0,56
8		Mainan	0,7	0,73
9		Fashion	0	0
10		Fashion	0	0
Rata-rata			0,564	0,608

Produk dengan skor tertinggi/kemiripan tertinggi dengan citra *query* akan muncul lebih dulu dibandingkan produk dengan skor yang lebih rendah. Kemiripan dilakukan dengan memanfaatkan tiga fitur yaitu fitur tekstual, fitur warna, dan fitur pencocokan objek menggunakan SURF. Akurasi hasil pencarian ini sangat bervariasi dari yang bisa banyak menghasilkan produk yang relevan dengan *precision@20* dan *recall* masing-masing 1, hingga tidak berhasil mencari produk yang relevan dengan *precision@20* dan *recall* masing-masing 0. Dari hasil penelitian citra produk yang kurang populer, tidak pernah diindeks oleh *search engine* memberikan hasil yang gagal, karena *search engine* memberikan *keywords/tags* yang terlalu umum.

Dari hasil penelitian yang telah dilakukan, saran yang dapat diberikan pada penelitian berikutnya dengan menggunakan *query expansion* sebagai salah satu alternatif untuk merekomendasikan kata kunci yang diharapkan. Selain itu, metode *keypoint* SURF hanya dapat mengenali citra yang memiliki beberapa fitur unik, sementara lemah jika mengenali gambar dengan warna yang atau tekstur yang homogen, sehingga kombinasi fitur tekstur dan warna dapat digunakan untuk membantu meningkatkan proses pencarian.

6. SUMBER PUSTAKA

BAY, H., ESS, A., TUYTELAARS, T. dan VAN GOOL, L., 2008. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3), hal.346–359.

HARSONO, H., 2016. *Indonesia will be Asia's next biggest e-commerce market*. [daring] TechCrunch. Tersedia pada: <https://techcrunch.com/2016/07/29/indonesia-will-be-asias-next-biggest-e-commerce-market/> [Diakses 12 Sep 2017].

HOU, J., ZHANG, D., CHEN, Z., JIANG, L., ZHANG, H. dan QIN, X., 2010. Web Image Search by Automatic Image Annotation and Translation. hal.3–6.

HSU, E., PAZ, C. dan SHEN, S., 2001. Clothing Image Retrieval for Smarter Shopping.

JUAN, L. dan GWUN, O., 2009. A Comparison of SIFT, PCA-SIFT and SURF. *International Journal of Image Processing (IJIP)*, 3(4), hal.143–152.

KIAPOUR, M.H., HAN, X., LAZEBNIK, S., BERG, A.C. dan BERG, T.L., 2015. Where to Buy It: Matching Street Clothing Photos in Online Shops. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE, hal.3343–3351.

LIU, Z., LUO, P., QIU, S., WANG, X. dan TANG, X., 2016. DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (1), hal.1096–1104.

PENA, M.G., 2012. *A Comparative Study of Three Image Matching Algorithms: SIFT, SURF, and FAST*. BiblioBazaar.

RUBNER, Y., TOMASI, C. dan GUIBAS, L., 2000. The Earth Mover's Distance as a Metric for Image Retrieval. *International Journal of Computer Vision*, 40(2), hal.99–121.

SURAL, S., QIAN, G., PRAMANIK, S. dan LANSING, E., 2002. Segmentation and Histogram Generation Using the HSV Color Space for Image Retrieval. In: *2002 International Conference on Image Processing (ICIP)*. hal.589–592.