IMPLEMENTASI ALGORITMA SIFT UNTUK MELAKUKAN KLASIFIKASI BAHAN BAKAR KENDARAAN RODA EMPAT PADA SPBU

Lingga Eka Pratama,

Teknik Informatika – Universitas Komputer Indonesia

Jl. Dipatiukur 112-114 Bandung
E-mail : linggaeka.p@gmail.com

ABSTRAK

Algoritma SIFT (*Scale Invariant Feature Transform*) merupakan salah satu algoritma yang berperan dalam pengenalan objek yang tahan dan efektif terhadap perubahan rotasi dan penskalaan. Penambahan metode *scan and matching* yang akan diuji seberapa efektif (*error* algoritma dan *error* sistem) kedua algoritma ini dapat mendeteksi dan melakukan objek berupa kendaraan roda empat sehingga secara tidak langsung dapat mengklasifikasikan bahan bakar pada SPBU.
Kata kunci : SIFT, *k-Nearest Neighbor*, Klasifikasi, *Keypoint*, Pendeteksian dan Pencocokan Objek

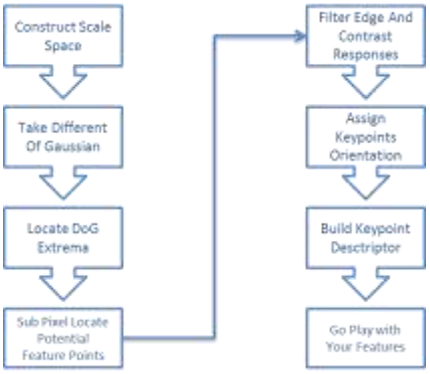
1. PENDAHULUAN

Diketahui bahwa algoritma SIFT telah banyak dipakai dan dilakukan pengujian [2], tetapi untuk melakukan kegiatan seperti pengenalan objek dan pencocokan data kemudian dilakukan klasifikasi sampai saat ini belum ada yang melakukannya. Berdasarkan kondisi dari bidang keilmuan, maka dalam penelitian ini dilakukan pengujian algoritma SIFT dengan penambahan metode *scan and matching* untuk melihat bagaimana kombinasi metode dan algoritma tersebut berjalan efektif dengan melakukan pencocokan data dan klasifikasi objek yang mana objek disini adalah kendaraan roda empat. Dilakukan pengujian ke kendaraan roda empat karena mengingat semakin bertambah dan berkembangnya pasar mobil pribadi di Indonesia sehingga perlu adanya sistem yang mampu mengklasifikasi mobil pribadi untuk pembatasan penggunaan bahan bakar di Indonesia.

2. ISI PENELITIAN

2.1 Algoritma SIFT

Algoritma SIFT (*Scale Invariant Feature Transform*) ini memiliki kemampuan deteksi dan pendeskripsian fitur fitur lokal di dalam data gambar dimana hasil dari pendeteksian fitur-fitur lokal tersebut dapat digunakan dalam penjejakan objek bergerak, selain itu metode ini memiliki daya tahan yang bagus dari segi skala, rotasi dan perubahan sudut pandang citra.



Gambar 1langkah-langkah algoritma SIFT

1. ***Scale Space Extrema Detection****:* tahapan pertama untuk pencari dan perhitungan Gambar dan skala. secara efisien diimplementasikan dengan fungsi *Different of Gaussian Scale Space* untuk melakukan identfikasi titik-titik potensi yang sesuai dengan orientasi dan skala invarian.

   Gaussian Blur

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e - (x^2 + y^2)/2\sigma$$

   dimana nilai sigma σ sudah ditentukan dan e = 2.71. selanjutnya proses

   Gaussian Scale Space :

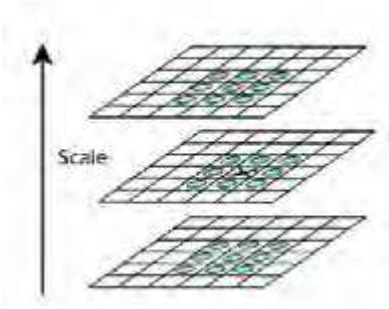
$$L(x,y,\sigma) = G(x,y,\sigma) * I(x,y)$$

   Dimana,
   $G(x,y,\sigma)$ = *Gaussian Blur*
   $I(x,y)$ = *original image*

   Deteksi ekstremum atau nilai maksimum dan minimum dilakukan dengan cara membandingan nilai setiap *pixel* pada DoG *scale space* dengan delapan *pixel* yang terdapat di sekelilingnya dan 9 *pixel* yang

bersesuaian pada citra DoG sebelum dan sesudahnya. Apabila nilai *pixel* lebih besar atau lebih kecil daripada nilai *pixel* pembandingnya maka koordinat *pixel* ditandai sebagai ekstremum. Proses ini bertujuan untuk mencari nilai potensi *keypoint*, seperti pada gambar 2 deteksi ekstremum

2. ***Keypoint Localization***: setiap titik calon lokasi, lokasi dan skala ditentukan oleh model pas. Ukuran dan stabilitas menjadi syarat untuk dipilih menjadi titik kunci (*keypoint*).



Gambar 2 deteksi ekstremum

3. ***Orientation Assignment***: satu atau lebih orientasi ditugaskan untuk lokasi setiap titik kunci (*keypoint)* sesuai dengan arah gambar gradien lokal. Semua operasi pada setiap data gambar akan di transformasi relatif pada orientasi yang telah ditetapkan, skala dan fitur setiap lokasi untuk memberikan invarian pada transformasi ini.

untuk *magnitude* persamaanya adalah:

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

L(x,y)	4	5	6
11	0.1347	0.1342	0.1327
12	0.1374	0.1352	0.1319
13	0.1376	0.1344	0.1301

Gambar 3 proses *magnitude*

Dimana nilai tengah berwarna merah merupakan *keypoint* dengan 8 titik tetangga yang ada di sekitar, lalu kemudian dihitung proses antara nilai atas dan bawah pada titik *keypoint* untuk kemudian ditambahkan nilai kanan dan kiri diantara nilai *keypoint*, dan hasilnya membentuk panjang dari arah orientasi *keypoint* tersebut.

L(x,y)	4	5	6
11		Magnitude 0.4789	
12		0.1352	
13			

Gambar 4 Hasil *Magnitude*

Selain nilai *magnitude*, ada juga nilai orientasi *keypoint*, yang menentukan arah dari nilai *keypoint*. Prosesnya adalah:

$$\theta(x,y) = \arctan\left(\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}\right)$$

L(x,y)	4	5	6
11	0.1347	0.1342	0.1327
12	0.1374	0.1352	0.1319
13	0.1376	0.1344	0.1301

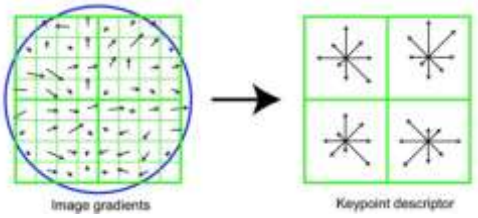
Gambar 5 Proses Orientasi

nilai merah berwarna pada posisi tengah pada gambar 5 merupakan nilai orientasi pada keypoint, lalu kemudian dihitung dengan persamaan orientasi sehingga hasilnya seperti pada gambar 6 di bawah ini.

L(x,y)	4	5	6
11	0.1347	0.1342	0.1327
12	0.1374	0.1352	0.1319
13	0.1376	0.1344	0.1301

Gambar 6 hasil proses orientasi

4. ***Keypoints Descriptor***: setiap deskriptor dihitung untuk setiap *keypoint*, lalu diubah untuk representasi yang memungkinkan tingkat signifikan pada setiap perubahan pencahayaan dan level distorsi bentuk atau gambar.



Gambar 7 *keypoint* deskriptor

2.2 Metode *Scan and Matching*

Pada proses Metode *scan and matching*, atau bisa disebut juga dengan *matching models* atau *matching descriptor*, ada beberapa cara yang dilakukan untuk proses pencocokan tersebut, *ofir pele* menyebutkan, sesuai dengan referensi dari *lowe* ada tiga cara yaitu dengan :

1. *Threshold on Distance = Bad Performance [1]*
2. *Nearest Neighbour = Better [1]*
3. *Ratio Test = best performance [1]*

Pada tiga cara diatas, disebutkan juga bagaimana cara-cara tersebut digunakan dan bagaimana hasil yang didapat. Untuk proses *matching* yang diambil pada penelitian ini, dilakukan proses *Nearest Neighbour* (jarak terdekat), dimana proses ini menggunakan deskriptor *keypoint* untuk kemudian diolah dengan cara L_2norm= *nearest neighbor*. Proses jarak terdekat ini, memanfaatkan nilai vektor deskriptor pada *keypoint* deskriptor *magnitude*, untuk kemudian diolah dan dilakukan perbandingan untuk mencari nilai rata-rata pada deskriptor dengan menggunakan persamaan (*mean)* rata-rata :

$$\frac{nilai\ data\ 1 + nilai\ data\ 2}{jumlah\ data} = rata - rata$$

Selain mencari rata-rata nilai vektor deskriptor, dilakukan juga proses perhitungan jarak terdekat (*nearest neighbor*) dimana proses ini melibatkan algoritma *k-Nearest Neighbor* sebagai proses perhitungannya. Pada proses perhitungan jarak terdekat pun, perhitungan nilai rata-rata pada nilai jarak yang didapat pun dilakukan, hal ini meminimalisir terjadinya proses *error matching.*

Proses *image matching* sendiri mempunyai nilai batas kemiripan, seperti yang dikatakan oleh *lowe*, bahwa nilai kemiripan objek pada proses *image matching* sendiri mempunyai batasan jika kurang dari nilai 0.25 pada proses nilai kemiripan, maka bisa dikatakan objek tersebut mirip atau sama.

2.3 Klasifikasi

Klasifikasi merupakan proses untuk menemukan model atau fungsi yang menjelaskan atau membedakan konsep atau kelas data, dengan tujuan untuk dapat memperkirakan kelas dari suatu objek yang labelnya tidak diketahui. Pada metode *scan and matching*, ada proses pencarian jarak terdekat *keypoint* menggunakan algoritma yang berhubungan dengan klasifikasi dan *clustering*.

Dalam klasifikasi data, ada beberapa metode atau algoritma yang digunakan untuk melakukan proses tersebut seperti *c4.5, rainforest, neural network* dan salah satunya adalah *k-nearest neighbor*. Pada penelitian kali ini, digunakan metode *k-nearest neighbor* dimana proses tersebut dilakukan untuk pencarian terhadap objek berdasarkan data pembelajaran yang jaraknya paling dekat dengan objek tersebut .

K-Nearest Neighbor merupakan metode yang bersifat *supervised*, dimana hasil dari *query instance* yang baru diklasifikasikan berdasarkan mayoritas kategori pada KNN.

Algoritma K-NN sendiri merupakan algoritma yang menentukan nilai jarak pada data *testing* dengan data training berdasarkan nilai terkecil pada nilai ketetanggan terdekat. Tujuan dari algoritma k-NN adalah untuk mengklasifikasi objek baru berdasarkan atribut dan *training samples*. Dimana hasil dari sampel uji yang baru diklasifikasikan berdasarkan mayoritas dari kategori pada k-NN.

Pada proses pencarian jarak terdekat, algoritma ini tidak menggunakan model apapun untuk dicocokkan dan hanya berdasarkan pada memori. Algoritma k-NN menggunakan jarak ketetanggaan sebagai nilai prediksi dari sampel uji yang baru. Jarak yang digunakan adalah jarak *Euclidean Distance*. Jarak *Euclidean* adalah jarak yang paling umum digunakan pada data numerik, dan rumusnya adalah :

$$d(a,b) = \sqrt{\sum_{r=1}^{n}(a_k - b_k)^2}$$

Dimana:

$d(a,b)$ = jarak *Euclidean* (*Euclidean distance*)
(a_k) = *record* ke- 1
(B_k) = *record* ke- 2
k = 1,2,3,…n

vektor	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0	0	0	0.0711	0.0711	0	0	0	0	0	0	2.1442	2.1442	0	0	0
2	0	0	0	0.0659	0.0659	0	0	0	0	0	0	0.0688	0.0688	0	0	0

$$d(vektor\ 1, vektor\ 2) = \sqrt{\sum_{a=1}^{16}(vektor\ 1_a - vektor\ 2_a)^2}$$



Gambar 8 Objek Pencocokan Data



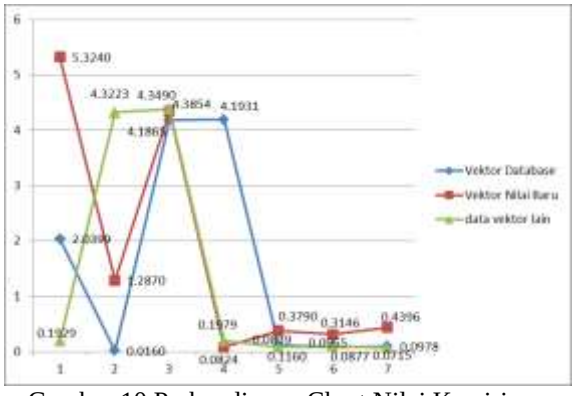
Gambar 9 Objek Pencocokan Data yang lain

Tabel 1 Nilai mobil honda jazz

vektor	jarak terdekat
1 Ke-2	2.0399
2 Ke-3	0.0160
3 Ke-4	4.1865
4 Ke-5	4.1931
5 Ke-6	0.1160
6 Ke-7	0.0877
7 Ke-8	0.0978

Tabel 2 Nilai Mobil Honda Jazz Lain

vektor	jarak terdekat
1 Ke-2	0.1929
2 Ke-3	4.3223
3 Ke-4	4.3854
4 Ke-5	0.1979
5 Ke-6	0.0839
6 Ke-7	0.0955
7 Ke-8	0.0715



Gambar 10 Perbandingan Chart Nilai Kemiripan

Pada gambar 10 yang terdapat diatas, dapat diketahui jika nilai kemiripan objek *honda jazz* pada perbandingna diatas memiliki nilai kemiripan yang lebih kecil yaitu 10.0432.

2.4 *Root Mean Square Error*

Perhitungan kesalahan merupakan pengukuran bagaimana nilai efektifivitas pencocokan data dan klasifikasi dilihat dari nilai *error* yang didapatkan, dapat dikatakan efektif jika nilai *error* proses pencocokan data dan klasifikasi terhitung kecil atau minor. Selisih antara dua nilai pencocokan data dan klasifikasi ditentukan dengan cara dihitung dengan suatu persamaan.

Sum square error (SSE) dapat dihitung dengan cara berikut :

1. Hitung nilai vektor data (nilai rata – rata menggunakan *Euclidean distance*) sebagai nilai pertama.
2. Hitung selisih antara nilai pertama dan nilai target (nilai vektor data pada *database*).
3. Kuadratkan setiap keluaran kemudian hitung seluruhnya.

   adapun rumusnya :

$$SSE_n =_v \sum\sum(T_{nv} - X_{nv})^2$$

   Dengan :
   T_{nv} = nilai vektor data pertama.
   X_{nv} = nilai target (nilai vektor data pada *database*).

Root Mean Square Error

1. Hitung *SSE*.
2. Hasilnya dibagi dengan perkalian antara banyaknya vektor pada suatu objek dan banyaknya nilai vektor yang digunakan.

   Persamaannya adalah:

$$RMS\ ERROR\ = \sqrt{\frac{SSE_n =_v \sum\sum(T_{nv} - X_{nv})^2}{n_p n_o}}$$

   Dengan :
   T_{nv} = nilai vektor data pertama.
   X_{nv} = nilai target (nilai vektor data pada *database*).
   n_p = jumlah vektor pada nilai keluaran.
   n_o = nilai vektor yang digunakan.

untuk proses *SSE* hasilnya adalah:

$$SSE_n =_v \sum\sum(T_{nv} - X_{nv})^2$$

SSE = $(12.7156 – 10.7370)^2$ = 3.9148 (nilai keluaran (*Toyota yaris*) – nilai target (*Honda jazz)*).
SSE = $(9.3494 – 10.7370)^2$ = 1.9254 (nilai keluaran (*Honda jazz*) – nilai target (*Honda jazz)*).

Setelah didapatkan nilai SSE (*sum square error*), proses selanjutnya adalah melakukan proses RMS (*root mean square error*) yaitu membagi nilai hasil SSE dengan total nilai vektor yang digunakan (tujuh nilai pada hasil proses *eucledian distance*) dan vektor yang diproses pada RMS dan SSE yaitu dua (vektor pada database dan vektor nilai keluaran).

$$RMS\ ERROR\ = \sqrt{\frac{SSE_n =_v \sum\sum(T_{nv} - X_{nv})^2}{n_p n_o}}$$

$$RMS\ ERROR = \sqrt{\frac{3.9148}{7*2}} = 1.0575$$ (*Toyota Yaris*)

$$RMS\ ERROR = \sqrt{\frac{1.9254}{7*2}} = 0.7416$$ (*Honda Jazz*)

Pada proses perhitungan diatas yaitu nilai kesalahan pada proses pencocokan data untuk *Toyota yaris* yaitu *1.0575* mempunyai nilai lebih besar dibandingkan nilai pencocokan data pada *Honda jazz 0.7416*, sehingga dapat disimpulkan bahwa proses pencocokan data terbilang efektif karena nilai kesalahan tidak terlalu besar pada proses pencocokan data.

2.5 Simulasi Pengujian Algoritma

Pengujian algoritma merupakan pengujian terhadap hasil pendeteksian dan pencocokan mobil pribadi dengan algoritma SIFT.

Pengujian dilakukan dua cara, cara pertama adalah melakukan pendeteksian kendaraan roda empat berdasarkan ukuran parameter *pixels* yang diuji untuk mendapatkan *keypoint*. Cara yang kedua adalah melakukan pencocokan data antara objek data *keypoint* yang berada di dalam *database* dengan beberapa objek yang sama namun ada perubahan rotasi dan perbedaan skala.

Tabel 3 Ukuran Parameter Pixels yang didapat

tabel ukuran parameter pixels		
No	**ukuran pixels**	**Rata-Rata**
1	10x7	null
2	15x10	null
3	30x20	null
4	45x30	4.479
5	60x40	6.563
6	68x45	7.387
7	75x50	-5.050
8	90x60	9.292
9	105x70	20.016
10	120x80	8.148
11	135x90	1.126
12	150x100	6.746
13	165x110	17.005
14	180x120	24.682
15	195x130	7.127
16	210x140	12.642
17	225x150	23.300
18	240x160	24.541
19	255x170	26.699
20	270x180	27.789
21	285x190	29.013
22	300x200	30.866

selain tabel pengujian parameter batas pendektesian algoritma SIFT diatas, adapun hasil dari pengujian nilai proses pencocokan data dengan perhitungan algoritma.

1. Melakukan pencocokan dengan Gambar yang di lakukan rotasi 180^0 derajat.



Gambar 11 Perbandingan Objek 1.

Gambar Awal : Gambar 8
Ukuran awal: 300 x 200 p
Gambar Pembanding : Gambar 4.11
Ukuran Pembanding : 300 x 200 p
Nilai Kemiripan : *7.0303*
Batas Nilai Kemiripan : 0.25
Nilai *Sum Square Error* : 2.1915
Nilai *Root mean Square Error* : 0.7912

2. Melakukan pencocokan dengan gambar yang mempunyai ukuran *pixels* yang berbeda.



Gambar 12 Perbandingan Objek 2.

Gambar: Gambar 4.8
Ukuran awal: 300 x 200 p
Gambar Pembanding: Gambar 4.12
Ukuran Pembanding: 45 x 30 p
Nilai Kemiripan: 8.5136
Batas Nilai Kemiripan: 0.25
Nilai *Sum Square Error*: 9.3593
Nilai *Root mean Square Error*: 1.6352

3. Melakukan pencocokan dengan gambar yang mempunyai ukuran *pixels* dan gambar yang berbeda.

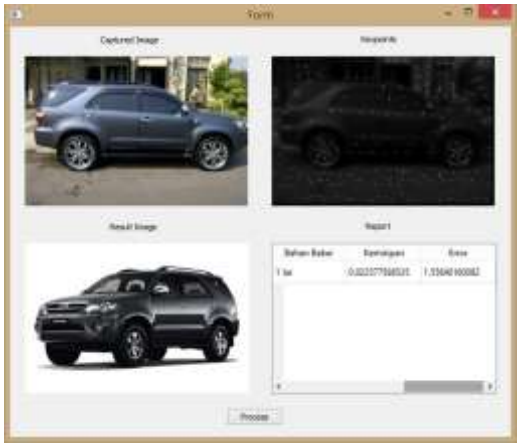


Gambar 13 Objek Pembanding 3

Gambar: Gambar 4.8
Ukuran awal: 300 x 200 p
Gambar Pembanding: Gambar 4.13
Ukuran Pembanding: 45 x 30 p
Nilai Kemiripan: 7.8198
Batas Nilai Kemiripan: 0.25
Nilai *Sum Square Error*: 19.7749
Nilai *Root mean Square Error*: 1.7514

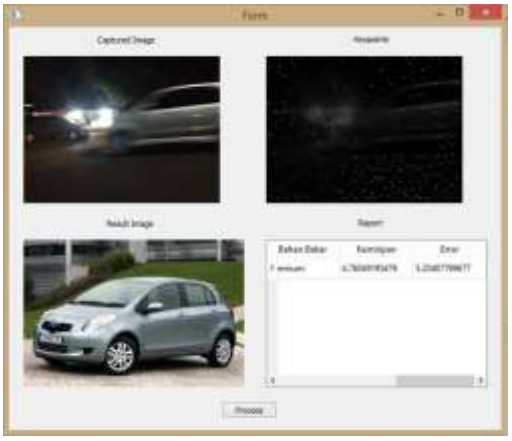
2.6 Hasil Pengujian Program

Selain pengujian algoritma diatas, adapun hasil pengujian sistem yang mana menunjukan proses dan hasil yang didapatkan, antara lain:



Gambar 14 Pengujian Program 1

Pada gambar 14, proses yang diujicobakan pada saat mobil berhenti, dan ternyata hasil yang didapat adalah data yang sesuai, dimana data yang ditangkap adalah *Toyota fortuner* dan hasil proses menunjukan *Toyota fortuner* dengan nilai kemiripan 0.02 dengan nilai kesalahan 1.5.



Gambar 15 Pengujian Program 2

Gambar 15 diatas ini, proses yang diujicobakan pada saat malam hari dengan tingkat pencahayaan yang kurang, dan ternyata hasil yang didapat adalah ketidak sesuaian data, dimana data yang ditangkap adalah Toyota avanza sedangkan hasil proses menunjukan Toyota yaris dengan nilai kemiripan 4.76 dengan nilai kesalahan 5.20.



Gambar 16 Pengujian Program 3

Pada gambar 16 diatas, objek yang ditangkap dirubah sudut posisi penangkapan objeknya, dan hasil yang didapat ternyata tidak sesuai, yang mana objek merupakan *Toyota avanza* sedangkan sistem menidentifikasi objek tersebut adalah *Suzuki karimun* dengan nilai kemiripan 0.01 dengan kesalahan 1.17.



Gambar 17 Pengujian Program 4

Pada pengujian 4, Objek yang ditangkap merupakan *Honda Jazz*, namun sistem menidentifikasi bahwa objek tersebut adalah *Toyota Fortuner* dengan nilai kemiripan 0.54 dengan nilai error 1.5.

3. PENUTUP

Dari kesimpulan yang didapat pada penelitian ini, algoritma SIFT dan metode *matching* mempunyai kekurangan dan kelebihan masing-masing. berikut adalah kesimpulan yang didapat pada penelitian ini, antara lain:

1. Implementasi sistem identifikasi dan klasifikasi pada penelitian ini secara fungsionalitas, belum bisa dikatakan efektif untuk digunakan pada objek bergerak, dikarenakan banyak terjadinya *loss error* (hasil tidak sesuai, banyak objek yang gagal ditangkap) dengan nilai kesalahan rata-rata 1.5 -5.00, sehingga masih perlu pengembangan lebih jauh lagi untuk sistem pada penelitian ini.
2. Untuk pengujian secara algoritma (perhitungan), algoritma SIFT dan metode *scan and matching*, bisa digunakan untuk proses identifikasi dan pencocokan objek dengan perhitungan secara algoritma, namun untuk pengujian secara fungsional sistem (aplikasi), masih belum bisa menangkap objek secara sempurna pada penelitian ini.
3. Dari segi kebutuhan, pada penelitian ini sistem belum bisa memenuhi kebutuhan klasifikasi dan pendeteksian kendaraan roda empat, dikarenakan sistem masih belum berjalan maksimal dan fungsionalitas masih belum bisa memenuhi kebutuhan klasifikasi.

SARAN

Saran untuk penelitian skripsi ini yaitu fungsionalitas sistem harus bisa berjalan sesuai kebutuhan dan diuji dengan banyak objek mobil pribadi, sehingga bisa terlihat hasilnya, selain itu pada segi implementasi perhitungan algoritma, perlu dipelajari kembali lebih lanjut dan dihitung secara manual serta analisisnya lebih bisa dikembangkan dan untuk metode *scan and matching* dengan proses yang lain (*ratio test*, dan *threshold on distance*) sehingga tercapai hasilnya dengan berbagai proses lain.

DAFTAR PUSTAKA

[1] [Lowe, David G., "*Distinctive Image Features from Scale-Invariant Keypoints*", International Journal Of Computer Vision, 60,2, pp. 91-100, 2004.

[2] Agustina Shanti Eka, Mukhlash Imam, 2012 "Implementasi Metode *Scale invariant Feature Transform* (SIFT) dan Metode *Continously Adaptive Mean-Shift (CAMSHIFT)* Pada Objek Bergerak", Fakultas Matematka dan Ilmu Pengetahuan Alam, Institut Teknologi Sepuluh November, 2012.

[3] Putra Darma, " Pengolahan Citra Digital" penerbit Andi, Indonesia, 2010.

[4] Jyoti joglekar[a], Shirish S.Gedam[b] , " *Image Matching With SIFT Features – A probabilistic Approach",* Centre of

Studies in Resources Engineering, IIT Bombay, Associate Professor, Mumbai, india, 2010.

[5] Lowe, David G., "*Local Feature View Clustering for 3D Object Recognition*", Computer Science Department, University of British Columbia, 2001.

[6] Tucker B. Allan, "*Second Edition Computer Science Handbook*" Chapman & Hall CRC , USA, 2004.

[7] Lowe, David G, "*Object recognition from local scale-invariant features*", *Proc. 7th International Conference on Computer Vision (ICCV'99)* (Corfu, Greece): 1150-1157, 1999

[8] Peffers K. Tuunanen T. Rothenberger M.A. Chatterjee S., "*A Design Science Research Methodology for Information Systems Research*" Published in Journal of Management Information Systems, Volume 24 Issue 3, Winter 2007-8, pp. 45-78, 2008.

[9] Olson E.B, "*Real-Time Correlative Scan Matching*", Department of Electrical Engineering and Computer Science, University of Michigan, 2009.

[10] A,bPulung Nurtantio Andono, aRicardus Anggi Pramunendar, aCatur Supriyanto, aGuruh Fajar Shidik, bI Ketut Eddy Purnama, bMochamad Hariadi, "*Enhancement Of 3d Surface Econstruction Of Underwater Coral Reef Base On Sift Image Matching Using Contrast Limited Adaptive Histogram Equalization And Outlier Removal*" a Faculty of Computer Science, Dian Nuswantoro University, bFaculty of Industrial Technology, Dept. of Electrical Engineering Institut Teknologi Sepuluh November, 2013.

[11] Away, Gunaidi Abdia, "*The Shortcut of Matlab Programming*" penerbit informatka, Bandung, 2006.