Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 2, No. 1, Januari 2018, hlm. 423-432 http://j-ptiik.ub.ac.id

Deteksi Objek Penghalang Secara *Real-Time* Berbasis Mobile Bagi Penyandang Tunanetra Menggunakan Analisis *Blob*

Achmad Jafar Al Kadafi[1], Fitri Utaminingrum[2]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]al.kadafi20@gmail.com, [2]f3_ningrum@ub.ac.id

Abstrak

Tunanetra merupakan suatu keadaan ketika indera penglihatan seseorang mengalami gangguan atau hambatan, sehingga membutuhkan alat bantu seperti tongkat ketika berjalan. Namun penggunaan tongkat tidak sepenuhnya membantu mereka dalam berjalan terutama untuk mendeteksi sebuah halangan. Dalam ilmu *computer vision* sangat memungkinkan para penyandang tunanetra mampu melakukan aktivitas berjalan seperti orang normal pada umumnya. Oleh karena itu, dalam penelitian ini dibangun sebuah sistem berbasis *computer vision* yang diterapkan pada sebuah perangkat *mobile* untuk mendeteksi halangan secara *real-time* ketika penyandang tunanetra berjalan didalam ruangan. Perangkat *mobile* akan dikondisikan pada ketinggian 1 meter diatas lantai dan sudut antara 52^o hingga 62^o untuk mendapatkan jarak sekitar 2 meter didepan pengguna. Secara umum proses deteksi halangan dibangun dengan menerapkan metode *Connected Component Labeling* untuk mendapatkan suatu *blob* dari citra. Untuk mendukung proses deteksi, tahap segmentasi dilakukan menggunakan metode *thresholding* dengan memanfaatkan model warna normalisasi RGB berdasarkan warna lantai yang dominan terang. Nilai *threshold* yang digunakan berdasarkan nilai minimal dan maksimal dari setiap komponen normalisasi RGB. Hasil pengujian menunjukkan bahwa sistem mampu mendeteksi halangan dengan tingkat akurasi sebesar 81.25%.

Kata kunci: *tunanetra, deteksi halangan, real-time, computer vision, normalisasi RGB, deteksi blob*

Abstract

Blind is a condition when the visual sense experiencing interference or obstacles, so requiring an aids like stick to walking. But, the using of stick does not help them much in walk especially to detect obstacles. In Computer Vision science so possible for people with the visual impairment can do an activity like normal people in general. In computer vision science is so possible for people with visual impairment can do walking activities like a normal people in general. Therefore, this research built a system based computer vision that is applied to a mobile device to detect obstacles on real-time when the visual impairment person walks indoors. Mobile devices will be conditioned at a height of 1 meter above the floor and angle between 52^o to 62^o to get a distance of about 2 meters in front of the user. In general, the obstruction detection process built by applying the Connected Component Labeling method to get a blob from the image. To support the detection process, segmentation process is done using threshold method by utilizing RGB normalization color model based on the dominant bright of floor color. The threshold value used is based on the minimum and maximum values of each component of RGB normalization. Test results shows that the system is able to detect obstacles with an accuracy of 81.25%.

Keywords: *blind, obstacle detection, real-time, computer vision, RGB normalization, blob detection*

1. PENDAHULUAN

Tunanetra merupakan suatu keadaan ketika fungsi indera visual seseorang (mata) mengalami kekurangan daya dalam melihat, sehingga dalam melakukan aktivitas dan berkomunikasi dengan lingkungan mereka menggunakan alat bantu atau menggunakan panca indera yang lain seperti pendengaran, penciuman, perasa (Grasianto dan Saphiranti, 2013; Ishartiwi, 2008). Menurut data Organisasi Kesehatan Dunia (WHO), pada tahun 2010 penyandang tunanetra telah mencapai 285 juta orang di seluruh dunia yang mana 14% dari angka tersebut telah mengalami kebutaan dan

sisanya dalam kondisi penglihatan lemah. Diperkirakan angka tersebut akan meningkat secara signifikan hingga tahun 2020. Ditambah lagi hampir 90% dari penyandang tunanetra tersebut tinggal di negara berkembang yang mana dukungan obat dan perawatan masih tergolong minim (Pascolini and Mariotti, 2011).

Salah satu permasalahan umum yang sering dialami oleh penyandang tunanetra yaitu ketika melakukan aktifitas berjalan. Ketika berjalan indera penglihatan sangat berperan penting dalam melakukan kontrol tubuh. Namun karena memiliki keterbatasan dalam penglihatan, kebanyakan penyandang tunanetra menggunakan alat bantu seperti tongkat sebagai pengganti indera penglihatan dan menggunakan indera pendengaran sebagai penerima respon dari tongkat tersebut. Untuk mengatasi masalah tersebut, berbagai kalangan dari berbagai bidang telah mencoba mengajukan beberapa solusi, seperti penerapan teknologi sebagai pengganti indera penglihatan.

Salah satu teknologi yang dapat dilakukan untuk mengatasi masalah ini adalah dengan memanfaatkan *computer vision*. *Computer vision* merupakan cabang dari ilmu komputer yang mempelajari bagaimana agar sebuah komputer mampu melihat layaknya manusia normal (Wahyudi dan Kartowisastro, 2011). Pada penelitian ini, penerapan *computer vision* difokuskan untuk mengidentifikasi objek berupa halangan secara *real-time* dengan memanfaatkan keadaan lantai. Halangan yang dimaksud adalah objek seperti orang, tiang, atau benda lain yang berada didepan penyandang tunanetra ketika berjalan yang dapat dikategorikan sebagai halangan.

Agar dapat membantu bagi penyandang tunanetra dalam melakukan aktifitas berjalan, maka dalam penelitian ini juga memanfaatkan perangkat *mobile* sebagai alat bantu utama karena memang saat ini perangkat *mobile* telah menjadi kebutuhan primer bagi setiap orang termasuk bagi penyandang tunanetra. Dalam hal ini perangkat *mobile* akan berperan sebagai perantara bagi penyandang tunanetra untuk memvisualisasikan keadaan lingkungan dengan memanfaatkan kelebihan yang dimilikinya berupa kamera. Karena perangkat *mobile* merupakan piranti yang memiliki keterbatasan dalam hal penyimpanan dan kecepatan komputasi (Zhong et al., 2013), dibandingkan dengan komputer atau *server*, maka pemilihan metode juga merupakan faktor penentu untuk mencapai hasil yang maksimal.

Dalam penelitian ini metode yang digunakan adalah *Connected Component Labeling* yang diterapkan untuk deteksi *blob*. *Blob* yang telah terdeteksi selanjutnya dianalisis untuk mendeteksi suatu halangan yang terdapat pada citra. Pada tahap awal, dilakukan proses segmentasi menggunakan metode thresholding dengan memanfaatkan model warna RGB yang telah dinormalisasi sebagai nilai *threshold*. Model warna ini digunakan untuk mendapatkan hasil yang maksimal dalam proses segmentasi dengan keadaan pencahayaan yang beragam, sehingga dapat mengurangi pengaruh intensitas cahaya dari luar (Adikara dkk., 2014).

Connected Component Labeling merupakan algoritme dasar dalam pengolahan citra digital yang secara umum digunakan dalam proses yang berhubungan dengan deteksi objek (Schwenk and Huber, 2015). Penggunaan metode *Connected Component Labeling* sendiri sudah sering dipakai untuk deteksi *blob*/objek. Seperti pada penelitian yang dilakukan oleh Rizki dkk (2010) untuk pencarian karakter plat dalam sistem pengenalan plat nomor kendaraan. Pada penelitian tersebut metode *Connected Componen Labeling* digunakan untuk melakukan proses pencarian plat nomor kendaraan, yang mana karakter pada citra disegmentasi tanpa harus melewati proses pencarian lokasi platnya terlebih dahulu. Hasilnya dengan melakukan evaluasi terhadap 20 sampel citra yang diambil pada berbagai kondisi menghasilkan nilai akurasi yang sangat baik, yaitu 85%.

Dari penjelasan tersebut, maka dalam penelitian ini dibangun sebuah sistem yang mampu mendeteksi objek secara *real-time* menggunakan analisis *blob*. Sistem akan diterapkan pada perangkat *mobile* dengan tujuan agar dapat digunakan secara efektif dan efisien bagi penyandang tunanetra. Diharapkan dengan adanya sistem ini, dapat mengurangi permasalahan yang sering dialami oleh penyandang tunanetra, terutama saat melakukan aktifitas berjalan, sehingga mereka dapat menjalani aktifitas layaknya orang normal.

2. LANDASAN TEORI

2.1 *Computer Vision*

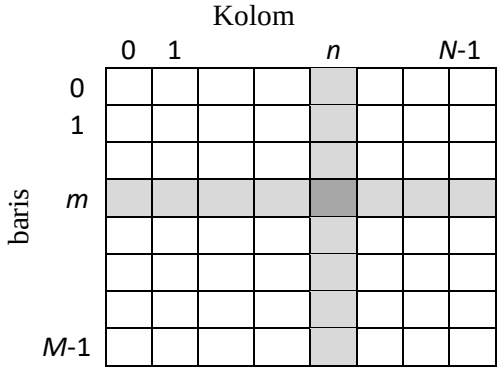
Computer vision merupakan cabang dari ilmu computer yang mempelajari bagaimana agar sebuah komputer mampu melihat layaknya manusia normal (Wahyudi dan Kartowisastro, 2011). Menurut Helmiriawan (2012), agar

mampu melihat layaknya seorang manusia, maka sebuah teknologi *computer vision* harus memiliki fungsi pendukung yang berjalan secara maksimal. Fungsi pendukung tersebut Antara lain:

a. Proses penangkapan citra/gambar (*image acquisition*)
b. Pengolahan citra (*image processing)*
c. Analisis data citra (*image analysis)*
d. Proses pemahaman citra (*image understanding)*

2.2 *Image Processing*

Pengolahan citra digital merupakan sebuah disiplin ilmu yang mempelajari tentang teknik-teknik mengolah citra. Sebuah citra digital dapat diwakili oleh sebuah matriks dua dimensi $f(x,y)$ yang terdiri dari *M* kolom dan *N* baris, yang mana perpotongan antara kolom dan baris disebut piksel (piksel= *picture element*) atau elemen terkecil dari sebuah citra (Kusumanto dkk., 2011; Kusumanto dan Tompunu, 2011). Gambar 1. merupakan bentuk representasi dari citra digital dalam 2 dimensi.



Gambar 1. Representasi citra digital 2 dimensi

2.3 Normalisasi RGB

Normalisasi RGB merupakan jenis lain dari model warna citra yang direpresentasikan menjadi 3 komponen warna yaitu r, g, b. Masing-masing komponen tersebut merepresentasikan prosentase dari sebuah piksel pada citra digital, sehingga jumlah keseluruhan dari komponen tersebut bernilai 1 (Kusumanto dan Tompunu, 2011; Adikara dkk., 2014). Menurut Adikara dkk (2014), normalisasi RGB dapat mengatasi pengaruh perbedaan intensitas cahaya pada objek yang sama. Persamaan normalisasi RGB dapat dituliskan pada Persamaan (1).

$$r = \frac{R}{R+G+B} \qquad (1)$$

$$g = \frac{G}{R+G+B}$$

$$b = \frac{B}{R+G+B}$$

R untuk nilai *red* pada suatu piksel, *G* untuk nilai *blue* pada suatu piksel, *B* untuk nilai *blue* pada suatu piksel dan *r, g,* dan *b* masing-masing untuk nilai normalisasi *R, G, dan B* dari piksel.

2.4 Segmentasi Citra

Segmentasi merupakan suatu proses untuk memisahkan antara objek satu dengan yang lain atau antara objek dengan *background* pada sebuah gambar, sehingga dapat digunakan sebagai *input* pada proses yang lain. Dengan proses tersebut sehingga didapatkan objek-objek dari sebuah citra yang dapat digunakan sebagai *input* pada proses yang lain (Murinto and Harjoko, 2009;Nugraheni, 2010;Senthilkumaran and Vaithegi, 2016). Secara sederhana teknik yang sering digunakan untuk memisahkan antara objek *background* dengan *foreground* adalah binerisasi atau *thresholding* (Fanani dkk., 2012). *Thresholding* merupakan sebuah teknik binerisasi yang proses binarisasinya dilakukan berdasarkan nilai *threshold* (batas ambang) (Fanani dkk., 2012). Nilai *threshold* dapat berupa informasi RGB, HSV, YCbCr dari suatu piksel, namun pada umumnya nilai yang sering digunakan adalah tingkat keabuan piksel. Suatu piksel dengan tingkat keabuan yang lebih tinggi dari *threshold* dikategorikan sebagai objek dan piksel yang lain dikategorikan sebagai *background.* Secara matematis metode *thresholding* dapat dituliskan dengan Persamaan (2).

$$g(x,y) = \begin{cases} 1 \; if \; f(x,y) \geq T \\ 0, \; if \; f(x,y) < T \end{cases} \qquad (2)$$

$g(x,y)$ adalah citra biner dari citra grayscale $f(x,y)$, dan T menyatakan nilai *threshold* (Kumaseh dkk., 2013; Senthilkumaran and Vaithegi, 2016).

Salah satu contoh model *thresholding* yang dapat digunakan adalah *color filtering. Color filtering* merupakan teknik *thresholding* yang dilakukan dengan menyaring suatu nilai warna tertentu menggunakan 2 batas, yaitu minimal dan maksimal (Amri dkk., 2014). Sebagai contoh adalah dengan memanfaatkan model warna HSV, yang mana setiap komponen warna HSV memiliki nilai *threshold* masing-masing, seperti pada Persamaan (3).

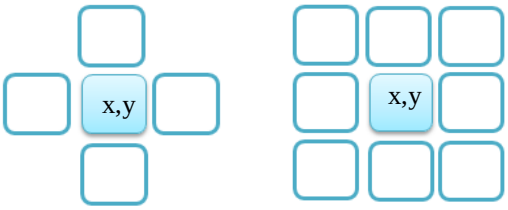
$$g(x,y) \begin{cases} 1, \text{if } (H_{f(x,y)} > Hmin \ \& \ H_{f(x,y)} > Hmax \ \& \ S_{f(x,y)} > Smin \ \& \ S_{f(x,y)} > Smax \ \& \ V_{f(x,y)} > Vmin \ \& \ V_{f(x,y)} > Vmax) \\ 0, \text{otherwise} \end{cases} \quad (3)$$

2.5 Deteksi objek

Deteksi objek merupakan sebuah tahap penting yang banyak diterapkan dalam proses pencarian gambar, *auto-annotation* gambar, dan pemahaman gambar (Wang et al., 2007). Menurut Wang et al (2007) pendekatan yang dapat digunakan untuk proses deteksi objek dikategorikan menjadi 2, yaitu *top-down* dan *bottom-up* atau kombinasi dari keduanya. Pendekatan *top-down* dilakukan dengan menyertakan tahap pelatihan untuk mendapatkan fitur-fitur kelas atau objek. Sedangkan pendekatan *bottom-up* dilakukan dengan menganalisis fitur gambar tingkat rendah atau menengah seperti tepi atau segmen. Menurut Schwenk and Huber (2015) pendekatan *botto-up* merupakan pendekatan yang secara umum digunakan dalam proses deteksi objek, yang mana algoritme yang digunakan adalah *Connected Component Labeling.*

Connected Component Labeling merupakan algoritme dasar dalam pengolahan citra digital yang secara umum digunakan dalam proses yang berhubungan dengan deteksi objek (Schwenk and Huber, 2015). Tujuan metode ini adalah untuk memberikan label yang sama pada piksel *foreground* yang saling berdekatan (biasanya dilakukan dengan unsur ketetanggaan (*connectivity*)), sehingga dapat diukur besar *foreground*/objek tersebut. *Foreground*/objek merupakan representasi nilai piksel dari sebuah *binary image*, biasanya dilambangkan dengan 2 nilai, yaitu 0 dan 255. Pada *binary image*, 2 piksel dapat disebut berdekatan apabila keduanya saling terhubung dan saling berwarna sama.

Menurut Schwenk dan Huber (2015), terdapat 2 tipe *connectivity* yang sering digunakan untuk membangkitkan label, yaitu *4-connectivity* dan *8-connectivity.* Pada unsur ketetanggaan *4-connectivity* suatu piksel pusat diberi label dengan memperhatikan 4 piksel *horizontal* dan *vertical* disekitarnya. Ketika ditemukan piksel tetangga dengan nilai model warna yang sama yang telah diberi label, maka piksel pusat akan diberi label sesuai label piksel tetangga tersebut. Berbeda pada unsur ketetanggaan *4-connectivity* suatu piksel pusat diberi label dengan memperhatikan 8 piksel *horizontal*, *vertical*, *diagonal* disekitarnya. Ketika ditemukan piksel tetangga dengan nilai model warna yang sama yang telah diberi label, maka piksel pusat akan diberi label sesuai label piksel tetangga tersebut. Bentuk tipe *connectivity* dapat dilihat pada Gambar 2.



(a) (b)

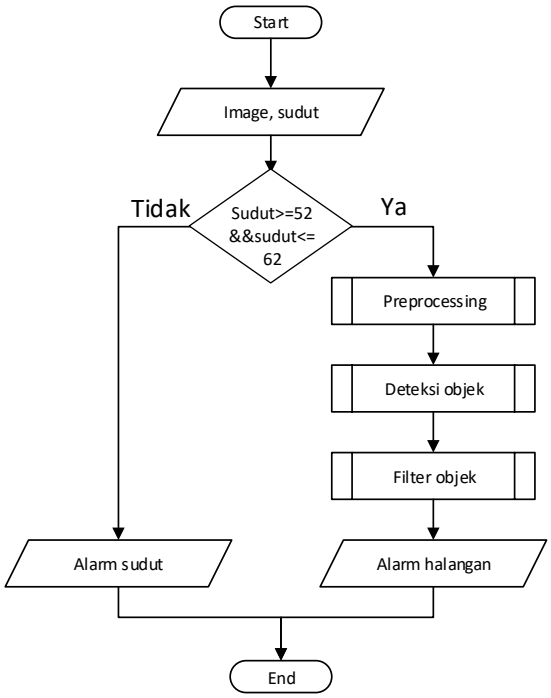
Gambar 2 Bentuk tipe *connectivity*
(a) Tipe 4-connectivity; (b) Tipe 8-connectivity

2.6 *Filtering* Citra

Filtering merupakan salah satu tahap dalam proses perbaikan citra, dengan tujuan untuk menghilangkan bagian-bagian tertentu yang tidak diinginkan dari sebuah data citra atau *noise* (Sholihin and Purwoto, 2014). Menurut Sholihin dan Purwoto (2014), proses *filtering* dapat dikategorikan menjadi 2, yaitu *spacial filter* dan *spacial frequency*. Proses *filtering* dengan *spacial filter* dilakukan dengan mengonvolusi suatu citra dengan citra lain, yang mana ukuran citra konvolusi lebih kecil dari citra yang dikonvolusi. Dengan menerapkan *spacial filter*, komputasi hanya berpengaruh terhadap sebuah piksel dan piksel-piksel tetangganya. Pada *spacial frequency*, proses pemfilteran dilakukan dengan mengkombinasikan nilai dari piksel sehingga membentuk suatu piksel tunggal.

3. PERANCANGAN SISTEM

Secara umum, proses sistem deteksi objek yang dikategorikan sebagai halangan berjalan dalam beberapa tahap, seperti yang disajikan pada Gambar 3. Ketika sistem berjalan, sistem akan menangkap gambar secara *real-time* sekaligus mengambil data sudut menggunakan sensor *accelometer* pada *mobile*. Ketika citra yang dihasilkan berada pada sudut yang sesuai, citra akan dianalisis pada tahap *preprocessing*, deteksi objek, dan filter objek untuk mendapatkan data objek yang dikategorikan sebagai halangan. Hasil analisis kemudian digunakan sebagai *output* sistem berupa suara.



Gambar 3 Diagram alir perancangan umum sistem

3.1 *Preprocessing*

Tahap *preprocessing* dilakukan dengan mengolah citra hasil tangkapan sedemikian rupa sehingga citra siap untuk diproses pada tahap selanjutnya. Proses yang dilakukan pada tahap *preprocessing* antara lain: pemotongan gambar (*cropping image*), *convert image*, dan segmentasi.

3.1.1 Cropping image

Pemotongan citra (*cropping image*) merupakan proses untuk membatasi bagian citra tertentu menjadi sebuah segmen yang digunakan sebagai *input* pada proses selanjutnya. Bagian citra yang dibatasi merupakan lokasi lingkungan nyata di depan pengguna sejauh 2 meter yang tampak pada citra, artinya bagian samping dan atas citra akan diabaikan. Dalam hal ini pembatasan dilakukan dengan memperhatikan jarak perangkat *mobile* dengan lantai, sudut perangkat *mobile* dan ukuran citra yang diambil. Dengan sudut perangkat *mobile* sebesar 52^o hingga 62^o dan ukuran citra sebesar 320 x 240 piksel, didapatkan nilai sebesar 160 piksel untuk batas atas segmen agar dapat mewakili jarak sejauh 2 meter. Langkah-langkah pada tahap *cropping image* adalah sebagai berikut:

1. Tentukan nilai *start x*, *start y*, batas *x* dan batas *y* dari citra
2. Buat garis *horizontal* dan *vertical* sesuai dengan nilai-nilai tersebut untuk menandakan *range*

3.1.2 Normalisasi RGB

Normalisasi RGB digunakan untuk mengubah setiap komponen nilai RGB normal menjadi nilai RGB yang telah dinormalisasi. Proses normalisasi dilakukan pada 40 citra data latih dengan menggunakan Persamaan (1). Nilai RGB yang telah di normalisasi kemudian dianalisa untuk mendapatkan nilai *mean*, *median*, dan min-max yang selanjutnya digunakan sebagai input pada proses segmentasi. Langkah-langkah pada tahap *convert image* adalah sebagai berikut:

1. Dapatkan nilai *R, G, B* piksel *x,y*
2. Hitung kembali nilai *R, G, B* dengan membagi nilai *R, G, B* dengan jumlah total *R, G, B* sesuai Persamaan (1)

3.1.3 Segmentasi citra

Proses segmentasi digunakan untuk membedakan antara *foreground* dengan *background.* Dalam penelitian ini metode segmentasi yang digunakan adalah *thresholding*, dengan nilai *threshold* yang digunakan adalah nilai batas atas dan bawah dari R (*red)*, batas atas dan bawah dari G *(Green)*, dan batas atas dan bawah dari B *(Blue)* yang telah dinormaliasi pada proses sebelumnya. Langkah-langkah pada tahap segmentasi adalah sebagai berikut:

1. Tentukan nilai *start x, start y*, batas *x* dan batas *y* dari citra sesuai pada tahap *cropping image*
2. Hitung nilai normalisasi RGB pada piksel *x,y*
3. Lakukan pengecekan untuk setiap nilai normalisasi RGB
4. Jika nilai normalisasi RGB termasuk dalam *range threshold* maka ubah warna piksel *x,y* dengan warna putih, dan jika tidak termasuk dalam range *threshold* maka ubah warna piksel *x,y* dengan warna hitam
5. Simpan hasil segmentasi pada *array*

3.2 Deteksi objek

Deteksi objek merupakan proses untuk mengidentifikasi objek-objek yang terdapat pada citra hasil segmentasi. Sebuah objek merupakan kumpulan dari piksel-piksel *foreground* yang saling berdekatan dalam range unsur 8 ketetanggaan. Proses deteksi dilakukan dengan menggunakan metode *Connected Component Analysis*, yang mana setiap objek yang telah teridentifikasi memiliki nilai label yang berbeda. Nilai tersebut akan digunakan untuk proses *filtering* pada tahap selanjutnya. Langkah-

langkah pada tahap deteksi objek adalah sebagai berikut:

1. Tentukan piksel pusat berdasarkan nilai piksel pada proses segmentasi
2. Dapatkan nilai piksel dari *array*
3. Lakukan pengecekan untuk label dan nilai piksel dari *array*
4. Jika label dan nilai piksel pusat bernilai 0, maka ubah warna piksel menjadi warna tertentu dan beri label pada piksel tersebut
5. Lakukan pengecekan tetangga dari piksel pusat dengan bentuk *8-connectivity*.
6. Lakukan pengecekan untuk piksel tetangga yang ditemukan
7. Jika label dan nilai piksel tetangga bernilai 0, maka ubah warna piksel menjadi warna sesuai piksel pusatnya dan beri label pada piksel tersebut
8. Ubah piksel tetangga menjadi piksel pusat
9. Lakukan kembali proses pengecekan mulai pada nomor 5 hingga seluruh piksel dilabeli
10. Hitung dan simpan jumlah piksel setiap label

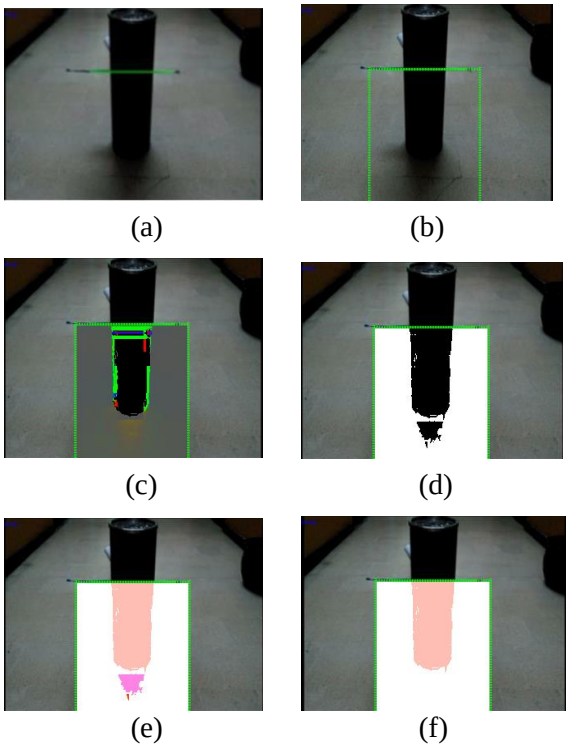
3.3 *Filtering* citra

Filter objek merupakan proses untuk memisahkan antara objek yang dikategorikan sebagai halangan dan bukan halangan. Parameter sebuah objek dapat dikategorikan sebagai halangan adalah ketika luas objek yang terdeteksi lebih dari 1054 piksel. Hasil dari proses filter objek merupakan sebuah nilai yang menunjukkan jumlah objek yang dikategorikan sebagai halangan. Langkah-langkah pada tahap filter objek adalah sebagai berikut:

1. Lakukan pengecekan jumlah piksel dari setiap label
2. Simpan label yang memiliki jumlah piksel lebih dari 1054 piksel.
3. Lakukan pengecekan label setiap piksel pada citra yang telah dibatasi pada proses *cropping image*
4. Jika label piksel tidak sama dengan label yang telah difilter, maka ubah warna piksel menjadi putih

4. IMPLEMENTASI

Pada penelitian ini sistem diimplementasikan pada perangkat *mobile* dengan fitur kamera *built-in*. Perangkat *mobile* yang digunakan memiliki spesifikasi prosesor Quad-core 1.2 GHz Cortex-A53, memori 1 Gb, ukuran layar 4,7 inchi, dan resolusi kamera 8 megapiksel. Implementasi algoritme yang digunakan untuk membangun sistem terdiri dari implementasi tahap awal sistem, implementasi tahap *preprocessing*, tahap deteksi objek, dan tahap filter objek. Contoh hasil implementasi algoritme dapat dilihat pada Gambar 4.



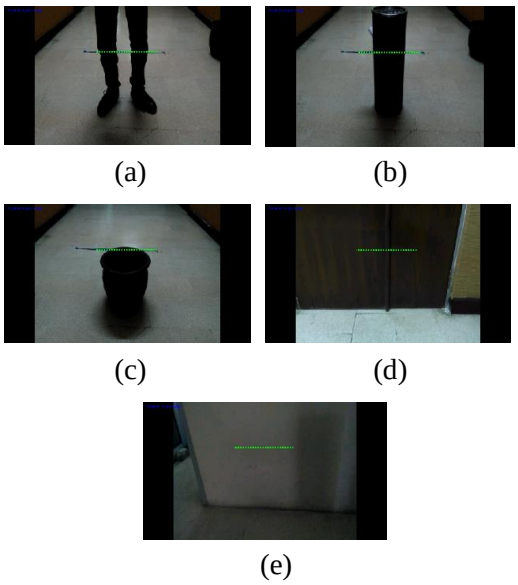
Gambar 4 Contoh hasil implementasi algoritme (a) citra awal; (b) *cropping image*; (c) normalisasi RGB; (d) segmentasi; (e) deteksi objek; (f) filter objek

Citra hasil proses *filtering* menunjukkan objek-objek yang dikategorikan sebagai halangan, seperti contoh pada Gambar 4f. Proses *filtering* ini merupakan tahap akhir dari proses deteksi halangan yang memberikan sebuah *output* berupa nilai/jumlah halangan yang terdeteksi. Ketika nilai yang di*return* lebih dari 0, sistem akan memberikan *output* berupa suara alarm halangan. Namun ketika nilai yang di*return* sama dengan 0, berarti sistem tidak mendeteksi adanya halangan dari citra yang diambil. Dalam kondisi ini sistem tidak akan memberikan *output* berupa suara namun suatu perintah untuk menghentikan suara-suara yang sedang berjalan. *Output* tersebut selanjutnya dapat digunakan oleh pengguna (penyandang tunanetra) sebagai penunjuk jalan.

5. PENGUJIAN

Pada penelitian ini pengujian dilakukan dengan 2 macam, yaitu pengujian nilai *threshold* dan pengujian akurasi deteksi dengan data uji sebanyak 80 citra. Citra untuk pengujian ini tidak diambil secara *realtime* namun dengan

mengambil beberapa *sample* citra dengan mengondisikan objek pada waktu dan jarak tertentu hingga menyerupai hasil tangkapan citra dalam mode *realtime.* Objek yang digunakan adalah manusia, tempat sampah, pot bunga, pintu, dan dinding. Untuk waktu dan jarak yang digunakan adalah pagi, siang, sore, malam hari dan jarak yang digunakan adalah 50 cm, 100 cm, 150 cm, dan 200 cm. Contoh citra untuk pengujian dapat dilihat pada Gambar 5.



(a) (b) (c) (d) (e)

Gambar 5. Contoh citra pengujian pada jarak 100 cm dan waktu malam hari
(a) objek manusia; (b) objek tempat sampah; (c) objek pot bunga; (d) objek pintu; (e) objek dinding

5.1 Pengujian nilai *threshold*

Citra hasil penangkapan perangkat *mobile* akan diujikan dengan menggunakan metode segmentasi *thresholding* dengan nilai batas ambang *(threshold)* yang berbeda-beda. Nilai *threshold* yang digunakan dalam pengujian ini adalah dengan menghitung nilai *mean*, *median*, dan nilai min-max pada masing-masing komponen RGB dari citra latih. Hasil pengujian adalah perbandingan tingkat akurasi untuk masing-masing nilai *threshold* terhadap jarak sesungguhnya dengan jarak hasil deteksi, yang mana setiap piksel mewakili 0.9375 cm pada jarak sebenarnya.

5.1.1 *Threshold* dengan nilai *mean*

Berdasarkan data hasil normalisasi RGB yang telah didapatkan, maka nilai *threshold* yang digunakan adalah Rmin = 0.319, Rmax = 0.349, Gmin = 0.337, Gmax = 0.367, Bmin = 0.296, dan Bmax = 0.334. Tabel 1 merupakan hasil pengujian dengan parameter nilai *mean*.

Tabel 1 Hasil pengujian parameter nilai *mean*

Hasil jarak yang diharapkan (cm)	Rata-rata akurasi deteksi jarak (%)			
	pagi	siang	sore	malam
50 cm	100%	100%	100%	100%
100 cm	62.36%	81.20%	54%	50%
150 cm	49.97%	69.73%	36.53%	62.26%
200 cm	56.78%	94.80%	25.50%	49.20%
Akurasi tiap waktu (%)	**67.28%**	**86.43%**	**54.01%**	**65.37%**
Rata-rata	**68.27%**			

Dari Tabel 1 dapat dilihat bahwa penggunaan parameter *threshold* dengan nilai *mean* mendapatkan nilai akurasi rata-rata sebesar 68.27%. Akurasi terburuk didapatkan pada waktu sore dan malam hari. Hal ini berarti proses segmentasi dengan menggunakan parameter ini belum mampu melakukan segmentasi dengan baik pada waktu sore dan malam hari. Namun pada waktu siang hari, parameter nilai ini mempunyai akurasi tertinggi diabandingkan waktu yang lain.

5.1.2 *Threshold* dengan nilai *median*

Berdasarkan data hasil normalisasi RGB yang telah didapatkan, maka nilai *threshold* yang digunakan adalah Rmin = 0.317, Rmax = 0.348, Gmin = 0.338, Gmax = 0.368, Bmin = 0.295, dan Bmax = 0.332. Tabel 2 merupakan hasil pengujian dengan parameter nilai median.

Tabel 2 Hasil pengujian parameter nilai *median*

Hasil jarak yang diharapkan (cm)	Rata-rata akurasi deteksi jarak (%)			
	pagi	siang	sore	malam
50 cm	100%	100%	100%	100%
100 cm	64.80%	78.20%	54%	53%
150 cm	69.60%	69.47%	36.67%	53.87%
200 cm	79.50%	96.00%	25.50%	49.00%
Akurasi tiap waktu (%)	**78.47%**	**85.92%**	**54.04%**	**63.87%**
Rata-rata	**70.57%**			

Dari Tabel 2 dapat dilihat bahwa penggunaan parameter *threshold* dengan nilai *median* mendapatkan nilai akurasi sebesar

70.57%. Nilai akurasi ini tidak jauh berbeda dengan nilai akurasi parameter *threshold* dari nilai *mean*. Akurasi terburuk untuk parameter nilai *median* juga didapatkan pada waktu deteksi sore dan malam hari. Hal ini juga berarti proses segmentasi dengan menggunakan parameter ini belum mampu melakukan segmentasi dengan baik pada waktu sore dan malam hari. Begitu halnya dengan parameter nilai *mean*, parameter nilai *median* mempunyai akurasi yang cukup baik pada waktu pagi dan siang hari.

5.1.3 *Threshold* dengan nilai min-max

Berdasarkan data hasil normalisasi RGB yang telah didapatkan, maka nilai *threshold* yang digunakan adalah Rmin = 0.285, Rmax = 0.374, Gmin = 0.322, Gmax = 0.403, Bmin = 0.264, dan Bmax = 0.366. Tabel 3 merupakan hasil pengujian dengan parameter nilai min-max.

Tabel 3 Hasil pengujian parameter *threshold* nilai min-max

Hasil jarak yang diharapkan (cm)	Rata-rata akurasi deteksi jarak (%)			
	pagi	siang	sore	malam
50 cm	80%	74%	80%	80%
100 cm	75.60%	56.60%	69%	69%
150 cm	88.13%	80.00%	86.53%	84.40%
200 cm	99.50%	99.50%	99.50%	85.40%
Akurasi tiap waktu (%)	**85.81%**	**77.42%**	**83.71%**	**79.60%**
Rata-rata	**81.64%**			

Dari Tabel 3 dapat dilihat bahwa penggunaan parameter *threshold* dari nilai min-max mendapatkan nilai akurasi sebesar 81.64%. Nilai akurasi ini cukup baik apabila dibandingkan dengan parameter nilai yang lain. Hal ini berarti parameter ini mampu melakukan proses segmentasi dengan baik dibandingkan dengan parameter *mean* atau *median*. Untuk penggunaan pada masing-masing waktu dapat dilihat bahwa parameter nilai min-max relatif baik. Namun penggunaan parameter ini mendapatkan nilai akurasi terendah pada waktu siang hari. Hal ini berbanding terbalik dengan parameter lain yang mana akurasi tertinggi didapatkan pada waktu siang hari.

5.2 Pengujian akurasi deteksi

Pengujian deteksi halangan dilakukan dengan menggunakan parameter *threshold* yang telah didapatkan pada pengujian segmentasi yaitu menggunakan min-max. Parameter ini memiliki nilai rata-rata akurasi tertinggi dibandingkan dengan parameter *threshold* yang lain pada masing-masing waktu yaitu sebesar 81.64%, sehingga nilai *threshold* yang digunakan adalah Rmin = 0.285, Rmax = 0.374, Gmin = 0.322, Gmax = 0.403, Bmin = 0.264, dan Bmax = 0.366. Tabel 4 merupakan hasil pengujian akurasi deteksi.

Tabel 4 Hasil pengujian akurasi deteksi

Jarak (cm)	Objek	Hasil yang diharapkan	Hasil deteksi				Akurasi
			pagi	siang	sore	malam	
50	1	√	√	√	√	√	100%
	2	√	√	√	√	√	100%
	3	√	√	√	√	√	100%
	4	√	√	√	√	√	100%
	5	√	X	X	X	X	0%
100	1	√	√	X	√	√	75%
	2	√	√	√	√	√	100%
	3	√	√	√	√	√	100%
	4	√	√	√	√	√	100%
	5	√	X	X	X	X	0%
150	1	√	√	X	√	√	75%
	2	√	√	√	√	√	100%
	3	√	√	√	√	√	100%
	4	√	√	√	√	√	100%
	5	X	X	X	X	X	0%
200	1	X	X	X	X	X	100%
	2	X	X	X	X	X	100%
	3	X	X	X	X	X	100%
	4	X	X	X	X	√	75%
	5	X	X	X	X	X	100%
Rata-rata akurasi							**81.25%**

Keterangan :

√ : terdeteksi
X : tidak terdeteksi

Dari Tabel 4 dapat dilihat bahwa dengan menggunakan parameter *threshold* nilai min-max, sistem mampu mendeteksi objek yang dikategorikan sebagai halangan dengan nilai akurasi sebesar 81.25%. Namun dengan menggunakan parameter *threshold* nilai min-max, nilai akurasi yang didapatkan untuk citra

objek nomor 5 sangat buruk, yaitu ketika objek yang digunakan adalah sebuah dinding berwarna putih. Hal tersebut dapat dilihat pada hasil pengujian citra nomor 5 untuk semua waktu pada jarak 50 cm, 100 cm, dan 150 cm menunjukkan bahwa hasil akurasi yang didapatkan 0%. Artinya bahwa untuk objek dengan komposisi warna yang dominan terang, sistem belum mampu mendeteksi halangan dengan baik.

6. KESIMPULAN

Berdasarkan hasil pengujian dari penelitian ini, maka dapat diambil beberapa simpulan sebagai berikut:

1. Sistem untuk deteksi halangan secara *realtime* dirancang dengan beberapa tahapan, yaitu tahap awal untuk deteksi sudut dan *preprocessing* citra yang meliputi tahapan *cropping*, normalisasi, dan segmentasi. Hasil dari proses *preprocessing* dapat digunakan sebagai *input* pada tahap selanjutnya yaitu deteksi objek dan *filtering* objek. Hasil tahap *filtering* merupakan suatu nilai yang digunakan sistem sebagai acuan untuk memberikan *output* yang digunakan penyandang tunanetra sebagai petunjuk berjalan.
2. Analisis *blob* diimplementasikan dalam beberapa tahap, yaitu tahap deteksi sudut, tahap *preprocessing* (*cropping*, *convert*, segmentasi*)*, deteksi objek dan filter objek. Sudut yang digunakan agar citra dapat diproses pada tahap selanjutnya adalah sebesar 52^o hingga 62^o. Pada tahap *preprocessing*, citra akan di*cropping* untuk mendapatkan range gambar yang sesuai. Data RGB hasil dari *cropping* akan dinormalisasi untuk proses segmentasi. Tahap segmentasi diimplementasikan menggunakan metode *thresholding* dengan nilai *threshold* yang digunakan adalah Rmin = 0.285, Rmax = 0.374, Gmin = 0.322, Gmax = 0.403, Bmin = 0.264, dan Bmax = 0.366. Untuk proses deteksi objek, sistem diimplementasikan dengan menggunakan metode *Connected Component Labelling*, yang mana objek yang dikategorikan sebagai halangan adalah dengan minimal luas objek sebesar 1054 piksel.
3. Masing-masing parameter *threshold* mempunyai nilai akurasi yang berbeda-beda. Untuk parameter *threshold* dengan nilai *mean dan median*, nilai akurasi yang didapatkan tidak terlalu berbeda yaitu sebesar 68.27% dan 70.58%. Pada kedua parameter tersebut akurasi terburuk didapatkan pada waktu sore dan malam hari. Sedangkan untuk parameter *threshold* dengan nilai min-max, didapatkan nilai akurasi tertinggi dari parameter *threshold* yang lain yaitu sebesar 81.64%. Namun penggunaan parameter ini tidak mendapatkan hasil akurasi tertinggi untuk waktu deteksi siang hari dibandingkan dengan parameter *mean* dan *median*.
4. Dari hasil pengujian akurasi deteksi dapat dilihat bahwa sistem yang dibangun mampu untuk mendeteksi halangan dalam kondisi *indoor* baik pada waktu pagi hari, siang hari, sore hari, maupun malam hari dengan nilai akurasi sebesar 81.25%. Namun pada kasus uji dengan objek halangan adalah sebuah dinding berwarna putih, sistem tidak mampu mendeteksi halangan dengan baik. Hal tersebut dikarenakan proses segmentasi menggunakan metode thresholding bergantung dengan komponen RGB dari citra.

7. DAFTAR PUSTAKA

Adikara, P.P., Rahman, M.A., Santosa, E., 2014. Pencarian Ruang Warna Kulit Manusia Berdasarkan Nilai Karakteristik (Λ) Matrik Window Citra. J. Teknol. Inf. Dan Ilmu Komput. JTIIK 1, 29–33.

Fanani, A., Prima, P., Hidayat, M.M., 2012. Local Thresholding Berdasarkan Bentuk Untuk Binerisasi Citra Dokumen. J. Ilm. Teknol. Inf. JUTI 10, 26–31.

Grasianto, I.D., Saphiranti, D., 2013. Pusat Pengembangan Kreativitas Anak Tunanetra. J. Tingkat Sarj. Bid. Senirupa Dan Desain 2, 1–6.

Ishartiwi, 2008. Mengenali Penyandang Tunanetra Dan Intervensi Pendidikannya.

Kumaseh, M.R., Latumakulita, L., Nainggolan, N., 2013. Segmentasi Citra Digital Ikan Menggunakan Metode Thresholding. J. Ilm. Sains 13, 74–79.

Kusumanto, R.., Tompunu, A.N., 2011. Pengolahan Citra Digital Untuk Mendeteksi Obyek Menggunakan Pengolahan Warna Model Normalisasi Rgb. Presented at the Seminar Nasional Teknologi Informasi & Komunikasi

Terapan 2011 (Semantik 2011).

Kusumanto, R.., Tompunu, A.N., Pambudi, W.S., 2011. Klasifikasi Warna Menggunakan Pengolahan Model Warna HSV. J. Ilm. Elite Elektro 2, 83–87.

Murinto, Harjoko, A., 2009. Segmentasi Citra Menggunakan Watershed Dan Itensitas Filtering Sebagai Pre Processing. Presented at the Seminar Nasional Informatika 2009, UPN "Veteran" Yogyakarta, pp. 43–47.

Nugraheni, M., 2010. Aplikasi Transformasi Watershed Untuk Segmentasi Citra Dengan Spatial Filter Sebagai Pemroses Awal. Presented at the Seminar Nasional Informatika 2010, UPN "Veteran" Yogyakarta, pp. 76–81.

Pascolini, D., Mariotti, S.P., 2011. Global estimates of visual impairment : 2010. Br. J. Ophthalmol. doi:10.1136/bjophthalmol-2011-300539

Schwenk, K., Huber, F., 2015. Connected Component Labeling Algorithm for very complex and high resolution images on an FPGA platform 9646. doi:10.1117/12.2194101

Senthilkumaran, Vaithegi, 2016. Image Segmentation By Using Thresholding Techniques For Medical Images. Comput. Sci. Eng. Int. J. CSEIJ 6, 1–13. doi:10.5121/cseij.2016.6101

Wahyudi, D.A., Kartowisastro, I.H., 2011. Menghitung Kecepatan Menggunakan Computer Vision. J. Tek. Komput. 19, 89–101.

Wardani, I.K., 2012. Strategi Presentasi Diri Pada Mahasiswa Tunanetra 1–18.

Zhong, Y., Garrigues, P.J., Bigham, J.P., 2013. Real Time Object Scanning Using a Mobile Phone and Cloud -based Visual Search Engine 20, 21– 23. doi:10.1145/2513383.2513443