p-ISSN 2301–4156 | e-ISSN 2460–5719

INVys: Sistem Navigasi Dalam Ruangan untuk Penyandang Tunanetra Menggunakan Kamera RGB-D

Widyawan[1], Muhammad Risqi Saputra[2], Paulus Insap Santosa[3]

[1,2,3] Departemen Teknik Elektro dan Teknologi Informasi, Fakultas Teknik, Universitas Gadjah Mada, Jl. Grafika No. 2 Kampus UGM, Yogyakarta 55281 INDONESIA (tel.: 0274-552305; fax: 0274-552305; email: [1]widyawan@ugm.ac.id, [3]insap@ugm.ac.id)
[2] Monash University, Indonesia, Green Office 9 Building, Jl. BSD Green Office Park, Sampora, Cisauk, Tangerang Regency, Banten 15345 INDONESIA (email: risqi.saputra@monash.edu)

[Diterima: 20 Februari 2023, Direvisi: 4 Oktober 2023]
Corresponding Author: Widyawan

INTISARI — Penelitian ini menyajikan sistem INVys untuk menyelesaikan masalah navigasi di dalam ruangan bagi penyandang tunanetra dengan memanfaatkan kemampuan kamera RGB-D. Sistem ini memanfaatkan informasi kedalaman yang disediakan oleh kamera untuk navigasi mikro yang melibatkan pengindraan dan penghindaran hambatan di lingkungan sekitar. Sistem INVys mengusulkan metode *auto-adaptive double thresholding* (AADT) baru untuk mendeteksi hambatan, menghitung jaraknya, dan memberikan umpan balik kepada pengguna agar dapat menghindari hambatan tersebut. AADT dievaluasi dan dibandingkan dengan metode *baseline* dan *auto-adaptive thresholding* (AAT) menggunakan empat kriteria, yaitu akurasi, presisi, ketahanan, dan waktu eksekusi. Hasilnya menunjukkan bahwa AADT unggul dalam akurasi, presisi, dan ketahanan, sehingga metode ini cocok digunakan untuk mendeteksi dan menghindari hambatan dalam konteks navigasi dalam ruangan bagi penyandang tunanetra. Selain navigasi mikro, sistem INVys memanfaatkan informasi warna yang disediakan oleh kamera untuk navigasi makro, yang dijalankan dengan mengenali dan mengikuti penanda navigasi yang disebut *optical glyphs*. Sistem ini menggunakan metode *automatic glyph binarization* untuk mengenali *optical glyphs* dan mengevaluasinya menggunakan dua kriteria, yakni akurasi dan waktu eksekusi. Hasil penelitian menunjukkan bahwa metode yang diusulkan akurat dan efisien dalam mengenali *optical glyphs*, sehingga cocok untuk digunakan sebagai penanda navigasi di lingkungan dalam ruangan. Lebih lanjut, penelitian ini juga menelaah korelasi antara ukuran *optical glyphs*, jarak *optical glyphs* yang dikenali, kondisi kemiringan *optical glyphs* yang dikenali, dan akurasi pengenalan *optical glyphs*. Korelasi ini menentukan ukuran *optical glyphs* minimum yang secara praktis dapat digunakan untuk navigasi dalam ruangan bagi penyandang tunanetra. Secara keseluruhan, penelitian ini menyajikan solusi yang menjanjikan untuk navigasi dalam ruangan bagi penyandang tunanetra dengan memanfaatkan kemampuan kamera RGB-D dan mengusulkan metode baru untuk mendeteksi dan menghindari hambatan serta mengenali penanda navigasi.

KATA KUNCI — Teknologi Bantu, Pengenalan Citra, Deteksi Objek, *Wearable Computer*.

I. PENDAHULUAN

Navigasi di dalam ruangan merupakan salah satu aktivitas yang sulit dilakukan bagi penyandang tunanetra, terutama jika mereka harus melakukannya secara mandiri. Mayoritas orang, termasuk penyandang tunanetra, menghabiskan sebagian besar waktunya di dalam ruangan, sehingga navigasi yang aman di lingkungan ini adalah suatu keharusan. Agar dapat bernavigasi dengan aman, penyandang tunanetra perlu melakukan dua aspek penting dalam pencarian arah, yaitu navigasi mikro dan navigasi makro [1], [2]. Navigasi mikro berhubungan dengan pengindraan lingkungan sekitar untuk mengetahui hambatan dan bahaya, sedangkan navigasi makro adalah navigasi ke tujuan yang jauh di luar lingkungan yang dapat dilihat/dirasakan secara langsung [3]. Selain itu, agar berhasil melakukan navigasi ke tujuan yang dikehendaki di dalam gedung, penyandang tunanetra harus dapat memastikan bahwa tempat di hadapan mereka merupakan tujuan yang benar. Tanpa bantuan orang lain atau alat bantu, penyandang tunanetra mungkin akan mengalami kesulitan dalam melakukan hal-hal ini.

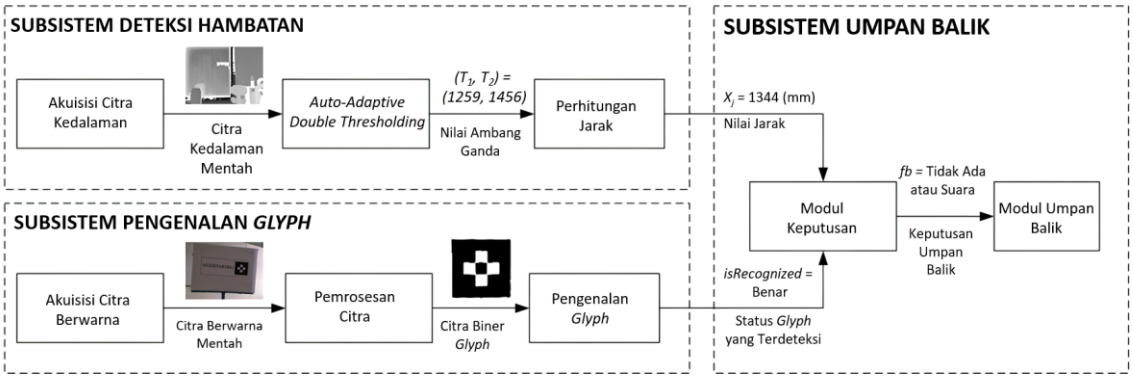
Untuk mengatasi masalah ini, secara konvensional, penyandang tunanetra menggunakan tongkat putih atau anjing pemandu. Tongkat putih memungkinkan penyandang tunanetra mendeteksi hambatan pada jarak rata-rata 1.500 mm [4], tetapi kegunaannya hanya terbatas pada aspek navigasi mikro. Anjing pemandu dapat membantu dalam kedua aspek pencarian jalan, tetapi anjing pemandu perlu dilatih dan biayanya cukup mahal bagi kebanyakan orang [5].

Banyak peneliti telah mengembangkan teknologi bantu (*assistive technology*, AT) dalam bentuk alat bantu perjalanan elektronik (*electronic travel aids*, ETA). Beberapa penelitian memadukan tongkat dengan laser [6], [7] atau teknologi ultrasonik [8], [9] untuk membantu penyandang tunanetra menghindari hambatan. Informasi terkait hambatan tersebut disampaikan kepada pengguna melalui suara atau getaran. Dilihat dari kemampuannya, alat ini lebih baik daripada tongkat putih konvensional karena secara otomatis dapat mendeteksi hambatan dan secara aktif memberikan umpan balik kepada pengguna. Namun, laser dan ultrasonik kurang dapat memberikan informasi terkait lingkungan sekitar [10] dan hanya mendukung fungsi navigasi mikro.

Di bidang navigasi makro, *global positioning system* (GPS) digunakan secara luas untuk memandu penyandang tunanetra saat berada di luar ruangan [11], [12]. Namun, GPS tidak cocok digunakan di dalam ruangan karena jangkauan sinyal satelit yang buruk dapat mengurangi akurasinya secara signifikan [13]. Banyak peneliti menggunakan label *radio-frequency identification* (RFID) [14], [15] sebagai penanda tempat di dalam ruangan. Akan tetapi, teknologi RFID bergantung pada infrastruktur yang perlu dipasang di dalam gedung oleh teknisi. Ketiadaan infrastruktur tersebut membuat bangunan menjadi tidak ramah bagi penyandang disabilitas.

Meskipun sebagian besar penelitian hanya berfokus pada satu aspek pencarian jalan (baik navigasi mikro maupun navigasi makro), beberapa peneliti telah mencoba menggabungkan kedua fungsi tersebut [16], [17]. Akan tetapi, integrasi berbagai macam teknologi (seperti laser, ultrasonik, GPS, atau RFID) yang dilakukan secara bersamaan berpotensi membuat sistem menjadi terlalu kompleks, tidak praktis,

p-ISSN 2301–4156 | e-ISSN 2460–5719



Gambar 1. Arsitektur INVys.

bergantung pada infrastruktur, dan memerlukan biaya perakitan yang mahal.

Akhir-akhir ini, kemajuan teknologi mengarah pada pengembangan kamera RGB-D berbiaya rendah, salah satunya adalah Microsoft Kinect. Tidak seperti kamera konvensional, kamera RGB-D menyediakan dua citra, yaitu citra kedalaman dari kamera kedalaman dan citra berwarna dari kamera RGB. Kamera jenis ini berguna untuk berbagai skenario dan aplikasi, termasuk navigasi dalam ruangan bagi penyandang tunanetra. Citra kedalaman dapat digunakan untuk memperkirakan jarak ke permukaan fisik terdekat dengan memberikan informasi kedalaman setiap piksel [18]. Sementara itu, citra berwarna dapat digunakan untuk fungsi navigasi makro dengan cara mengenali objek atau penanda tercetak yang ditempatkan di titik-titik utama.

Penelitian ini mengembangkan sistem navigasi dalam ruangan bagi penyandang tunanetra, yang disebut INVys, berdasarkan kamera RGB-D. INVys menggunakan metode *auto-adaptive double thresholding* (AADT) baru untuk mendeteksi hambatan dan mengirimkan umpan balik terkait hambatan tersebut kepada pengguna. AADT membagi area citra kedalaman dan mencari nilai dua ambang terbaik secara otomatis.

Di bidang navigasi makro, INVys menggunakan *optical glyphs* sebagai penanda tercetak untuk menandai titik-titik penting. Penelitian ini menyempurnakan metode pengenalan *optical glyphs* milik Kirillov [19] dengan menggunakan *automatic glyph binarization*.

Bagian selanjutnya dari makalah ini disusun sebagai berikut. Bagian II menjelaskan navigasi mikro dan penelitian terkait. Bagian III memberikan gambaran umum arsitektur sistem INVys. Bagian IV menjelaskan fungsi penghindaran hambatan dan Bagian V menjelaskan algoritma untuk mengenali *glyph*. Pengaturan dan hasil eksperimen disajikan di Bagian VI. Akhirnya, Bagian VII adalah kesimpulan.

II. NAVIGASI MIKRO

Navigasi mikro tampaknya menarik lebih banyak minat dan publikasi dibandingkan navigasi makro dalam bidang bantuan navigasi bagi penyandang tunanetra. Salah satu teknologi navigasi mikro yang paling awal dan menonjol adalah laser, mulai dari tongkat laser [20] hingga tongkat putih virtual [21], dan masih banyak lagi yang mendeteksi hambatan dan objek berbahaya [22], [23]. Metode triangulasi umumnya digunakan untuk menghitung jarak hambatan menggunakan pengukuran laser dengan cara menghitung sudut sinar pantul yang menyebar melewati lensa penerima [24]. Peneliti lain [25] menggabungkan laser inframerah dan Wiimote untuk menemukan posisi pengguna di sebuah ruangan. Namun, pendekatan ini memerlukan infrastruktur yang mahal. Selain itu, pengukuran jarak menggunakan laser memerlukan banyak energi [26].

Perangkat ultrasonik banyak digunakan untuk menghindari hambatan dengan cara menggabungkannya dengan tongkat [6], [9], perangkat *wearable* [16], robot/sistem bergerak [17], atau bahkan robot pemandu [27]. *Time of flight* (ToF) sebagian besar digunakan untuk menghitung jarak hambatan dengan mengukur interval waktu pengiriman sinyal dan penerimaan kembali gema. Akan tetapi, teknologi laser dan ultrasonik tidak dapat memberikan informasi memadai tentang lingkungan [28]. Selain itu, dengan menggunakan kedua teknologi tersebut, sistem perlu menggabungkan lebih dari satu pengirim/penerima sensor laser/ultrasonik untuk memetakan pemandangan lengkap di depan pengguna, sehingga dapat membuat sistem menjadi sangat kompleks, padahal kesederhanaan dan portabilitas merupakan aspek penting dalam mengembangkan ETA bagi para penyandang tunanetra [29].

Berbeda dengan penelitian sebelumnya, penelitian ini menggunakan kamera kedalaman, sehingga dapat secara langsung melihat tampilan lengkap peta pemandangan di depan pengguna hanya dengan satu sensor dan tanpa perangkat tambahan. Dengan menggunakan kamera kedalaman, sistem ETA dapat menjadi lebih sederhana dan mudah digunakan. Namun, untuk mendeteksi hambatan, data kedalaman mentah dari citra kedalaman harus diproses lebih lanjut. Beberapa peneliti telah mengembangkan metode untuk mendeteksi hambatan dari citra kedalaman. Metode *baseline* diimplementasikan dengan mempertimbangkan tinggi dan lebar penyandang tunanetra untuk menemukan titik terjauh yang dapat dijangkau pada citra kedalaman [30]. Pendekatan lain diterapkan dengan membagi citra kedalaman menjadi beberapa area dan memproses setiap area secara terpisah untuk mengurangi kompleksitas pemandangan [5].

Metode lain menggabungkan informasi dari citra kedalaman dan citra berwarna dan memproses data menggunakan algoritma pemrosesan citra [31], [32] atau visi komputer [28], [33] untuk melakukan penghindaran hambatan. INVys menerapkan metode AADT dengan menggabungkan prinsip pembagian citra kedalaman menjadi beberapa area berdasarkan kebutuhan rekomendasi arah yang mudah dipelajari/dimengerti oleh penyandang tunanetra dan analisis berbasis histogram kedalaman yang menghasilkan penghindaran hambatan yang cepat, efektif, dan kuat.

Label RFID sering digunakan sebagai penanda ruangan atau arah tertentu untuk memandu penyandang tunanetra

berpindah dari satu ruangan ke ruangan lain [15]. Meskipun pemanfaatan teknologi RFID ini dianggap efektif untuk memandu para penyandang tunanetra, RFID merupakan teknologi yang bergantung pada infrastruktur, sehingga banyak label RFID yang harus dipasang di dalam sebuah bangunan. Alih-alih menggunakan teknologi RFID, INVys memanfaatkan kamera berwarna untuk mengenali penanda tercetak.

III. ARSITEKTUR SISTEM

INVys mengandalkan perangkat lunak yang dapat memproses data dari kamera RGB-D untuk melakukan fungsi navigasi mikro dan makro. Sistem ini dibagi menjadi tiga subsistem, seperti yang ditunjukkan pada Gambar 1. Subsistem pertama, yaitu deteksi hambatan, menggunakan modul AADT untuk mendeteksi hambatan terdekat dan menghitung jaraknya dalam milimeter. Subsistem kedua, pengenalan *glyph*, memproses citra berwarna untuk menemukan dan mengklasifikasikan *glyph*. Subsistem ketiga, umpan balik, menggunakan modul keputusan untuk menentukan perlu tidaknya umpan balik diberikan kepada pengguna berdasarkan data dari dua subsistem pertama. Jika umpan balik diperlukan, modul umpan balik mengirimkan informasi dalam bentuk suara.

IV. DETEKSI HAMBATAN

A. PERSYARATAN DASAR

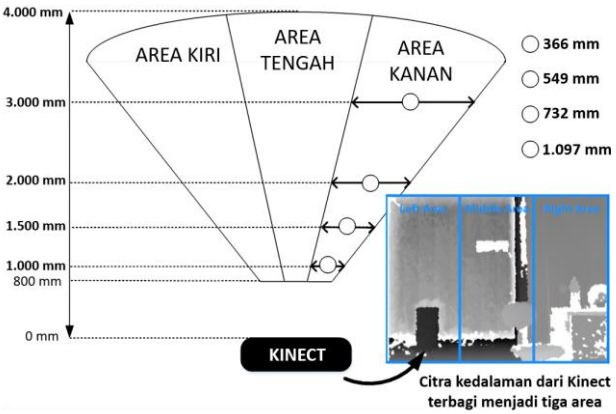
Aspek "keramahan" merupakan salah satu hal yang menentukan kematangan sistem deteksi dan penghindaran hambatan [29]. Untuk memenuhi kebutuhan tersebut, penelitian ini membagi area di hadapan penyandang tunanetra menjadi tiga, yang masing-masing mewakili jalur yang dapat dilalui, yaitu area tengah (lurus depan), area kiri, dan area kanan. Dengan dibagi menjadi tiga area, penyandang tunanetra hanya perlu mengingat tiga arah, sehingga memudahkan mereka mengikuti instruksi sistem.

Gambar 2 mengilustrasikan implementasi pembagian area menjadi tiga subarea. Dengan metode ini, setiap subarea dari citra kedalaman dapat sepenuhnya mencakup objek yang besar, seperti tubuh manusia. Sebagai contoh, mengacu pada Gambar 2, pada jarak 1,5 meter dari Kinect, setiap subarea dari citra kedalaman dapat mencakup objek dengan lebar maksimum 549 mm. Angka ini masih lebih tinggi jika dibandingkan dengan rata-rata lebar bahu orang Singapura dan Indonesia [34].

B. PENDEKATAN BASELINE

Citra kedalaman berisi informasi kedalaman setiap piksel dalam milimeter. Jadi, pada dasarnya, citra ini dapat digunakan untuk mendeteksi hambatan. Mengingat citra kedalaman memiliki resolusi sebesar 640 × 480 dan sensor yang mampu merekam 30 fps, jumlah total sampel kedalaman yang dianalisis mencapai lebih dari sembilan juta sampel. Permasalahan yang timbul adalah cara membuat data yang sangat banyak tersebut menjadi lebih bermakna bagi penyandang tunanetra.

Pendekatan *baseline* dikembangkan dengan membagi citra kedalaman menjadi banyak area dan menghitung nilai jarak rata-rata untuk setiap area [5]. Untuk meningkatkan efisiensinya, citra kedalaman 640 × 480 dibagi menjadi 32 × 40 blok dan rata-rata nilai piksel dihitung. Selanjutnya, semua blok dikelompokkan menjadi area 5 × 3, lalu sepuluh nilai tengah dalam setiap area dipilih setelah proses penyortiran. Selanjutnya, nilai rata-rata terakhir dari jarak hambatan dihitung berdasarkan nilai-nilai tersebut. Pendekatan ini menghasilkan 15 nilai jarak (satu untuk setiap wilayah) yang digunakan sebagai metrik hambatan.



Gambar 2. Area di depan penyandang tunanetra dibagi tiga.

C. AUTO-ADAPTIVE THRESHOLDING (AAT)

Auto-adaptive thresholding (AAT) adalah versi awal algoritma untuk mendeteksi hambatan, yaitu metode AADT. AAT mengacu pada metode yang secara otomatis dapat menghasilkan nilai ambang yang spesifik untuk subarea citra kedalaman. Ambang digunakan untuk memisahkan objek terdekat dan objek lain di belakangnya. AAT dimulai dengan membagi citra kedalaman menjadi tiga area, mengubah setiap area menjadi histogram kedalaman, dan menemukan dua puncak terdekat pada histogram tersebut [35]. Nilai ambang untuk setiap area citra kedalaman ditentukan secara otomatis dengan menerapkan metode *Otsu thresholding* [36] pada area di antara dua puncak terdekat di histogram kedalaman. Metode *Otsu thresholding* menghasilkan nilai ambang terbaik dengan memaksimalkan keterpisahan antara dua kelas yang dihasilkan oleh nilai ambang. Dengan demikian, data di antara dua puncak terdekat dibagi menjadi dua kelas, C_0 dan C_1, dengan memaksimalkan varians antarkelas menggunakan (1).

$$\sigma^2(k^*) = arg\ max(\sigma^2(k)), 1 \le k \le L \quad (1)$$

dengan $\sigma^2(k^*)$ adalah nilai ambang k^* yang memaksimalkan varians antarkelas, $\sigma^2(k)$ merupakan varians antarkelas untuk setiap nilai ambang k, dan k adalah setiap kemungkinan nilai ambang yang ada dalam rentang 1 dan nilai maksimum L pada citra. Varians antarkelas itu sendiri dihitung menggunakan (2).

$$\sigma^2 = \omega_0(\mu_0 - \mu_T)^2 + \omega_1(\mu_1 - \mu_T)^2 \quad (2)$$

dengan σ^2 adalah varians antarkelas, ω_0 adalah probabilitas kelas C_0, ω_1 adalah probabilitas kelas C_1, μ_0 adalah rata-rata kelas C_0, μ_1 adalah rata-rata kelas C_1, dan μ_T adalah keseluruhan rata-rata citra.

Berdasarkan percobaan sebelumnya, diketahui bahwa AAT efektif untuk mendeteksi dan menghitung jarak hambatan dengan jarak kurang dari 2.500 mm [35]. Jika hambatan berada lebih jauh dari jarak tersebut, akurasi AAT mulai menurun. Hal ini terjadi karena AAT tidak dapat membedakan antara objek dan lantai di depannya ketika lantai terdeteksi pada jarak lebih dari 2.500 mm.

D. AUTO-ADAPTIVE DOUBLE THRESHOLDING (AADT)

AADT adalah pengembangan lebih lanjut dari AAT. AADT dikembangkan untuk memecahkan permasalahan utama AAT, yaitu membedakan objek dengan lantai di depannya ketika objek berada pada jarak lebih dari 2.500 mm. Tidak seperti AAT yang hanya menggunakan satu nilai ambang, AADT menghasilkan dua nilai ambang untuk membedakan hambatan dan lantai di depannya. AADT menghasilkan nilai

p-ISSN 2301–4156 | e-ISSN 2460–5719

ambang ganda yang bervariasi untuk setiap area citra kedalaman (adaptif) secara otomatis. Perbedaan utama antara AADT dan AAT adalah cara memilih puncak pada histogram, cara menghasilkan ambang, dan cara menghitung nilai jarak akhir. Proses lengkap metode AADT dijelaskan sebagai berikut.

1) AKUISISI CITRA KEDALAMAN

INVys memperoleh data kedalaman mentah dari Kinect dan mengubah data tersebut menjadi citra *grayscale* 8 bit untuk divisualisasikan di layar menggunakan (3).

$$i_n = 255 - \left(\frac{255-(max(d_n-800,0))}{3.200}\right) \quad (3)$$

dengan i_n merupakan piksel ke-n pada citra *grayscale* dan d_n merupakan informasi kedalaman ke-n pada citra kedalaman.

2) MEMBAGI CITRA KEDALAMAN MENJADI TIGA AREA

Berdasarkan persyaratan dasar yang dijelaskan pada bagian sebelumnya, citra kedalaman dibagi menjadi tiga area: kiri, tengah, dan kanan. Frekuensi kemunculan langkah ini dan sesudahnya adalah 2 Hz. Semua langkah untuk setiap subarea selanjutnya diproses secara terpisah dan bersamaan.

3) *DOWNSAMPLING*

Untuk meningkatkan efisiensi dan mempercepat proses komputasi, *downsampling* dilakukan dengan hanya menggunakan satu nilai piksel untuk setiap blok piksel berukuran 2 × 2. Pengambilan sampel untuk setiap 2 × 2 piksel dianggap cukup untuk mengurangi data yang diproses, dengan tetap mempertahankan informasi citra yang cukup.

4) KONVERSI HISTOGRAM KEDALAMAN

Data yang telah melalui *downsampling* di langkah sebelumnya dikonversi menjadi histogram kedalaman dengan mengelompokkan setiap piksel ke dalam seratus kelompok. Citra direpresentasikan sebagai distribusi piksel dengan interval 40 mm antara setiap kelompok (karena cakupan kamera kedalaman adalah 0-4.000 mm dan dibagi menjadi seratus kelompok).

5) DETEKSI DAN PEMILIHAN PUNCAK

Ketika informasi kedalaman dikonversi menjadi histogram kedalaman, histogram ini akan penuh dengan puncak dan lembah yang merepresentasikan maksimum dan minimum lokal. Maksimum lokal dalam histogram kedalaman umumnya menunjukkan sebuah objek [35]. Oleh karena itu, objek terdekat ditentukan dengan menentukan maksimum lokal terdekat dalam histogram kedalaman. Proses ini dimulai dengan menghitung semua nilai kontras untuk setiap kelompok dalam histogram kedalaman menggunakan (4).

$$kontras(i,n) = \sum_{k=i-n}^{i+n} P_k - \sum_{k=i-2n}^{i-n-1} P_k - \sum_{k=i+n+1}^{i+2n} P_k \quad (4)$$

dengan i adalah posisi yang diamati, n adalah parameter untuk menjumlahkan kontras puncak dan tetangganya, serta p_k adalah nilai pada posisi k dalam histogram kedalaman. Angka n digunakan untuk menyaring derau dan posisi puncak lokal yang tidak terduga [37].

Setelah semua nilai kontras diperoleh, proses dilanjutkan dengan memilih nilai kontras yang dapat merepresentasikan maksimum lokal. Sebagai maksimum lokal, nilai kontras pada histogram memiliki nilai positif. Proses pemilihan dilakukan dengan mengurutkan nilai kontras dari yang tertinggi ke yang terendah dan memilih kelompok yang sesuai dengan kriteria berikut. Pertama, nilai kontras pada posisi ke-i harus lebih besar dari 300. Jika nilainya di bawah 300, histogram pada posisi tersebut tidak cukup curam untuk dianggap sebagai maksimum lokal yang independen. Kedua, jarak antar maksimum lokal harus lebih dari empat kelompok histogram; jika tidak, akan dianggap sebagai bagian dari maksimum lokal yang lain. Angka-angka tersebut ditentukan berdasarkan pengamatan. Setelah menemukan semua maksimum lokal, sistem memilih nilai yang paling dekat dengan Kinect (puncak/maksimum lokal dengan jumlah kelompok terkecil).

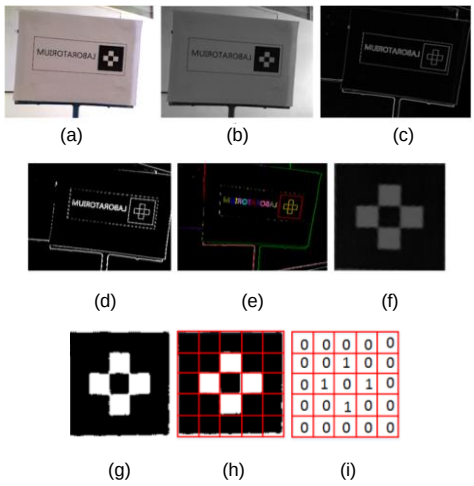
6) PEMILIHAN AMBANG GANDA

Langkah selanjutnya adalah menemukan dua nilai ambang (T_1, T_2) yang akan mengecualikan puncak terdekat dari data lain di sekitarnya. Nilai ambang pertama (T_1) adalah minimum lokal pertama, dan nilai jaraknya (x) lebih kecil daripada nilai jarak puncak terdekat. Nilai ambang kedua (T_2) sama dengan nilai ambang pertama, tetapi memiliki nilai jarak (x) yang lebih besar dari puncak terdekat. Menemukan minimum lokal dari maksimum lokal memerlukan perhitungan kemiringan setiap titik data tetangga di sekitar maksimum lokal dan berhenti ketika kemiringannya sangat berubah. Ambang adalah nilai jarak pada titik ketika gradien berubah secara dramatis. Proses rinci untuk menentukan nilai ambang pertama dan kedua ditunjukkan pada Algoritma 1 dan Algoritma 2.

Algoritma 1 Menentukan Nilai Ambang Pertama
1: $i = p$ dengan y_p adalah puncak terdekat dalam histogram kedalaman
2: x_i = kelompok data ke-i
3: y_i = frekuensi informasi kedalaman dalam kelompok ke-i
4: T_1 = nilai ambang pertama
5: d = jarak antara dua kelompok
6: $j \leftarrow 0$
7: **while** $i > 0$ **do**
8: slope $\leftarrow (y_i - y_i\text{-}1)/(x_i - x_i\text{-}1)$
9: **if** slope ≤ 0 **then**
10: **if** $y_i\text{-}j < (y_i/2)$ **then**
11: $T_1 \leftarrow y_i\text{-}j \times d$
12: **end if**
13: **end if**
14: $j \leftarrow j + 1$
15: $i \leftarrow i - 1$
16: **end while**

Algoritma 2 Menentukan Nilai Ambang Kedua
1: $i = p$ dengan y_p adalah puncak terdekat dalam histogram kedalaman
2: x_i = kelompok data ke-i
3: y_i = frekuensi informasi kedalaman dalam kelompok ke-i
4: n = jumlah kelompok dalam histogram kedalaman
5: T_2 = nilai ambang kedua
6: d = jarak antara dua kelompok
7: $j \leftarrow 0$
8: **while** $i < n$ **do**
9: **if** $i = n\text{-}1$ **then**
10: $T_2 \leftarrow i \times d$
11: **end if**
12: $slope \leftarrow (y_{i+1} - y_i)/(x_{i+1} - x_i)$
13: **if** $slope \geq 0$ **then**
14: **if** $y_{i+j} < (y_i/2)$ **then**
15: $T_2 \leftarrow y_{i+j} \times d$
16: **end if**
17: **end if**
18: $j \leftarrow j + 1$
19: $i \leftarrow i + 1$
20: **end while**

Berdasarkan kedua algoritma tersebut, selanjutnya ditambahkan persyaratan minimum untuk menentukan ambang, yaitu frekuensi informasi kedalaman pada nilai ambang harus kurang dari setengah frekuensi informasi kedalaman pada puncak yang diamati ($< y_i/2$). Hal ini dilakukan untuk menghindari ditemukannya minimum lokal yang terlalu dekat dengan puncak karena minimum lokal dapat menjadi bagian dari puncak.



Gambar 3. Contoh hasil proses pengenalan *glyph*: (a) citra RGB yang diakuisisi, (b) citra *grayscale*, (c) citra tepi, (d) citra biner, (e) citra *blob*, (f) citra *glyph* yang terdeteksi dalam *grayscale*, (g) *glyph* yang terdeteksi dalam citra biner, (h) proses *glyph binarization*, (i) hasil *glyph binarization*.

7) PENGHITUNGAN JARAK AKHIR

Penghitungan jarak akhir dilakukan dengan menghitung nilai rata-rata informasi kedalaman yang ada antara ambang pertama (T_1) dan ambang kedua (T_2) dengan menggunakan (5).

$$x_j = \sum_{k=1}^{n} \frac{i_k}{n}, T_1 < i_k < T_2 \quad (5)$$

dengan x_j adalah jarak rata-rata dari objek terdekat pada area citra kedalaman j, i_k adalah nilai kedalaman pada posisi piksel k pada area citra kedalaman j, T_1 adalah nilai ambang pertama, T_2 adalah nilai ambang kedua, dan n adalah jumlah i_k. Hasil dari proses ini adalah tiga nilai jarak objek terdekat (dalam milimeter), masing-masing nilai jarak objek untuk setiap area citra kedalaman.

V. PENGENALAN *GLYPH*

Optical glyph adalah penanda tercetak yang direpresentasikan dengan *grid* persegi yang baris dan kolomnya dibagi rata. Setiap sel diisi dengan warna hitam atau putih, kecuali baris/kolom pertama dan terakhir yang hanya diisi dengan warna hitam.

Optical glyph umumnya digunakan dalam *augmented reality* untuk menghasilkan objek 3D tepat di atas *glyph* yang dikenali. Teknik ini diadaptasi ke dalam penelitian ini dengan cara mengenali *glyph*, sehingga dapat memberikan informasi kepada para penyandang tunanetra tentang rambu-rambu navigasi, tempat, atau titik-titik penting lainnya. Metode yang digunakan untuk mengenali *glyph* didasarkan pada metode yang dikembangkan oleh Andrew Kirillov [19], yang telah dimodifikasi serta ditambahkan proses *glyph binarization*. Metode ini dimulai dengan mengakuisisi citra berwarna dari kamera RGB-D. Resolusi dan *frame rate* dari kamera berwarna adalah 640 × 480 dan 30 fps. Setelah akuisisi citra berwarna, langkah selanjutnya dibagi menjadi dua langkah utama, yaitu pemrosesan citra dan langkah pengenalan *glyph*. Kedua langkah tersebut dijalankan oleh modul pengolahan citra dan modul pengenalan *glyph*.

A. PEMROSESAN CITRA

Tujuan dari langkah ini adalah memproses citra berwarna yang diperoleh dan menemukan *optical glyph* yang berpotensi untuk dikenali. Langkah ini dijelaskan sebagai berikut.

1) KONVERSI KE CITRA *GRAYSCALE*

Dengan menggunakan rekomendasi ITU-BT.709, citra RGB yang diperoleh dikonversi ke dalam citra *graysacle* untuk memudahkan pemrosesan. Rekomendasi ini mengalikan nilai merah-hijau-biru dari piksel pemrosesan dengan bobot tertentu, yaitu 0,2125 untuk merah, 0,7154 untuk hijau, dan 0,072 untuk biru.

2) DETEKSI TEPI

Langkah selanjutnya adalah menemukan semua tepi citra *grayscale* yang diperoleh dari langkah sebelumnya. Tujuan langkah ini adalah untuk menemukan semua tepi yang membentuk persegi panjang, sehingga sistem dapat memastikan bahwa persegi panjang tersebut merupakan sebuah *glyph*. Metode yang digunakan untuk menemukan tepi objek adalah dengan mencari nilai selisih maksimum antara piksel-piksel tetangga pada empat arah di sekitar piksel yang diproses. Langkah ini dilakukan dengan menggunakan (6).

$$p' = max(|p_1 - p_5|, |p_2 - p_6|, |p_3 - p_7|, |p_4 - p_8|) \quad (6)$$

dengan p' adalah nilai selisih akhir, p_1 adalah piksel tetangga kiri-atas, p_2 adalah piksel tetangga tengah-atas, p_3 adalah piksel tetangga kanan-atas, p_4 adalah piksel tetangga tengah-kanan, p_5 adalah piksel tetangga kanan-bawah, p_6 adalah piksel tetangga tengah-bawah, p_7 adalah piksel tetangga kiri-bawah, dan p_8 adalah piksel tetangga tengah-kiri.

3) DETEKSI *BLOB*

Untuk menemukan *blob*, citra tepi pertama-tama dikonversi menjadi citra biner menggunakan ambang. Selanjutnya, *blob* dideteksi menggunakan algoritma pelabelan komponen terhubung di atas citra biner. Pelabelan komponen terhubung memindai semua piksel pada citra biner dan mengategorikannya ke dalam kelompok berdasarkan nilai konektivitasnya [38].

4) DETEKSI PERSEGI PANJANG

Setelah kelompok piksel yang terhubung ditemukan, setiap kelompok dianalisis untuk menemukan kelompok yang merupakan persegi panjang. Langkah-langkah lengkapnya meliputi penentuan kotak batas piksel-piksel yang terhubung; pencarian pusat kotak batas; pendeteksian sudut pertama dari piksel-piksel yang terhubung dengan cara mencari titik terjauh dalam piksel-piksel yang terhubung dari pusat kotak batas; pendeteksian sudut kedua dari piksel-piksel yang terhubung dengan cara mencari titik terjauh dalam piksel-piksel yang terhubung dari sudut pertama, pendeteksian sudut ketiga dan keempat dengan cara mencari dua titik terjauh dari garis lurus yang terbentuk dari sudut pertama dan sudut kedua; serta pengecekan semua garis yang terhubung oleh keempat sudut tersebut agar telah berada dalam batas distorsi. Batas distorsi ini digunakan untuk memastikan bahwa piksel-piksel yang terhubung dalam pemrosesan membentuk objek segi empat. Batas distorsi (dm) dihitung dengan menggunakan (7).

$$dm = max\left(m, \left(r \times \frac{w+h}{2}\right)\right) \quad (7)$$

dengan m adalah distorsi minimum yang diizinkan, r adalah batas distorsi relatif, w adalah lebar kotak pembatas, dan h adalah tinggi kotak pembatas. Hasil dari dm akan dibandingkan dengan jarak rata-rata dari sisi pemrosesan persegi panjang dengan pusat kotak batas.

5) DETEKSI GLYPH

Pada langkah selanjutnya, semua persegi panjang yang terdeteksi diperiksa untuk memastikan bahwa persegi panjang tersebut adalah *glyph*. Langkah ini dilakukan dengan membandingkan kecerahan rata-rata antara area piksel di dalam dan di luar persegi panjang. Perbedaan kecerahan tersebut sangat tinggi mengingat batas *glyph* berwarna hitam

p-ISSN 2301–4156 | e-ISSN 2460–5719

dan selalu dikelilingi oleh area putih. Selanjutnya, sistem hanya memotong area citra yang berisi *glyph*.

6) KONVERSI KE CITRA BINER MENGGUNAKAN *OTSU THRESHOLDING*

Proses terakhir dari langkah pengolahan citra adalah mengubah citra yang sudah dipotong menjadi citra biner dengan menggunakan *Otsu thresholding*. Dengan demikian, hasil langkah ini adalah citra yang terdiri atas dua kelas, yaitu kelas objek, yang biasanya ditandai dengan warna putih, dan kelas latar belakang, yang ditandai dengan warna hitam. Contoh hasil proses ini dapat dilihat pada Gambar 3.

B. PENGENALAN

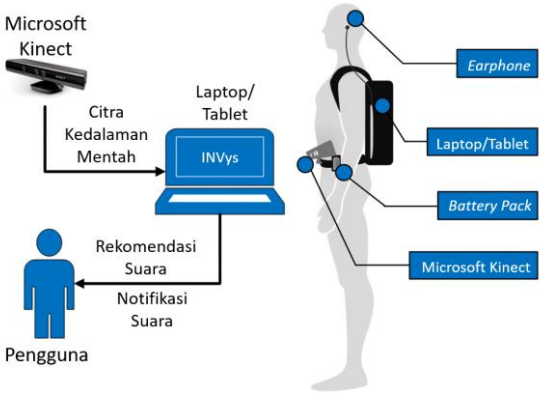
Langkah-langkah pengenalan dibagi menjadi dua proses. Langkah pertama disebut *glyph binarization*. *Glyph binarization* mengubah *glyph* yang terdeteksi menjadi nilai biner 0 dan 1. Langkah ini dilakukan dengan membagi *glyph* ke dalam jumlah baris dan kolom yang sama dan menghitung jumlah piksel berwarna (putih atau hitam) dari setiap sel. Jika sebuah sel didominasi oleh piksel hitam (lebih dari 60%), sel tersebut akan berubah menjadi 0. Sebaliknya, jika sel tersebut penuh dengan piksel putih (lebih dari 60%), sel tersebut akan terisi dengan 1. Hasil dari *glyph binarization* adalah sebuah matriks yang hanya memiliki nilai 0 dan 1.

Metode *glyph binarization* Kirillov memiliki kelemahan, yaitu ukuran *glyph* harus ditentukan terlebih dahulu. Oleh karena itu, pengguna harus membuat *glyph* sesuai dengan ukuran yang telah ditentukan. Penelitian ini mengatasi kelemahan tersebut dengan mengusulkan *automatic glyph binarization*. *Automatic glyph binarization* mendeteksi ukuran *glyph* secara otomatis berdasarkan kategori ukuran *glyph*. Pada umumnya, *glyph* yang digunakan berukuran 5 × 5 hingga 10 × 10. Ukuran *glyph* yang lebih besar dari 10 × 10 jarang digunakan karena sulit dibuat tanpa perangkat lunak komputer. *Automatic glyph binarization* akan memeriksa bahwa *glyph* yang terdeteksi memenuhi salah satu ukuran *glyph*, yaitu ukuran 5 × 5 hingga 10 × 10. Metode pendeteksian didasarkan pada karakteristik *glyph*, yaitu data putih (nilai = 1) akan selalu dimulai dari baris/kolom kedua.

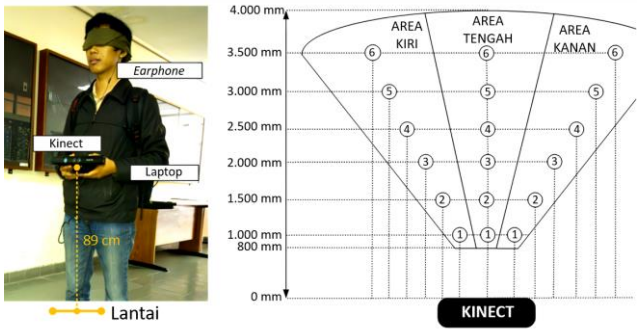
Proses pendeteksian ukuran dimulai dengan membagi *glyph* ke dalam matriks 10 × 10 (s = 10) dan memeriksa bahwa data putih (nilai = 1) berada (lebih dari 60%) di baris kedua. Jika sudah berada di baris kedua, berarti ukuran *glyph* sudah sesuai. Dengan demikian, ukuran *glyph* ditentukan sebagai 10 × 10. Jika tidak ditemukan *glyph* dengan ukuran tersebut, proses dilanjutkan dengan membagi *glyph* menjadi ukuran yang lebih kecil, yaitu 9 × 9 (s = 9). Rincian proses pendeteksian ukuran *glyph* ditunjukkan pada Algoritma 3.

Algoritma 3 Pendeteksian Ukuran *Glyph*

1: $I(w,h)$ = sebuah citra *glyph* I, dengan w lebar and h tinggi
2: $I_{m,n}$ = piksel pada kolom ke-m dan baris ke-n dalam citra I
3: s = ukuran *glyph* ($s \times s$)
4: cw_s = lebar tiap sel dalam *glyph* $s \times s$
5: ch_s = tinggi tiap sel dalam *glyph* $s \times s$
6: *result* = ukuran terakhir *glyph* terdeteksi
7: $s \leftarrow 10$
8: **while** $s \geq 5$ **do**
9: $cw_s \leftarrow w / s$
10: $ch_s \leftarrow h / s$
11: **for** $i \leftarrow 1$ to $s - 2$ **do**
12: r ← 0
13: **for** $a \leftarrow 1$ to ch_s **do**
14: $n \leftarrow ch_s + a$
15: **for** $b \leftarrow 1$ to cw_s **do**



Gambar 4. INVys terdiri atas empat bagian utama yang akan digunakan oleh para penyandang tunanetra (gambar di sebelah kanan). Aliran data di antara setiap bagian ditunjukkan pada gambar di sisi kiri.



(a) (b)

Gambar 5. Implementasi, (a) seorang pria dengan mata tertutup sedang melakukan percobaan, (b) posisi objek dalam eksperimen pendeteksian rintangan.

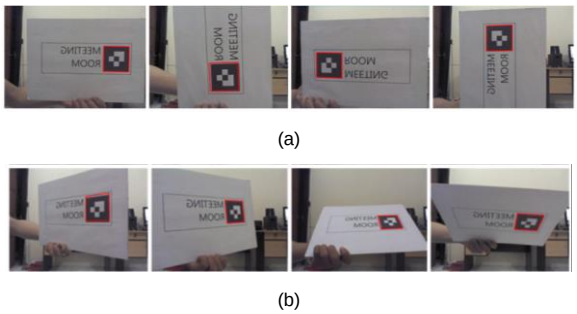
16: $m \leftarrow (cw_s \times i) + b$
17: **if** $I_{m,n}$ sama dengan warna putih **then**
18: $r \leftarrow r + 1$
19: **end if**
20: **end for**
21: **end for**
22: **if** $r > (0.6 \times cw_s \times ch_s)$ **then**
23: *result* ← s
24: **break**
25: **end if**
26: **end for**
27: $s \leftarrow s - 1$
28: **end while**

Langkah kedua adalah pencocokan *glyph*. Pada langkah ini, matriks dari langkah sebelumnya dibandingkan dengan data yang tersimpan dalam basis data. Pencocokan *glyph* dilakukan dalam empat kondisi yang memungkinkan, yaitu kondisi ideal (*glyph* tidak dirotasi), dirotasi 90°, dirotasi 180°, dan dirotasi 270°. Pencocokan dilakukan dengan cara memutar matriks untuk setiap kondisi, mengulang setiap nilai pada matriks, dan membandingkan nilai tersebut dengan matriks yang dihasilkan dari penyimpanan citra dalam basis data. Contoh hasil dari proses binerisasi dan pencocokan *glyph* ditunjukkan pada Gambar 3(h) dan Gambar 3(i).

VI. PERCOBAAN

Percobaan dilakukan untuk mengevaluasi dua fungsi utama sistem INVys, yaitu penghindaran hambatan sebagai fungsi navigasi mikro dan pengenalan tulisan sebagai fungsi navigasi makro. Pengaturan percobaan dan hasilnya dikelompokkan ke dalam dua kelompok. Konfigurasi perangkat keras, pengaturan eksperimen, indikator kinerja, dan hasil dijelaskan sebagai berikut.

p-ISSN 2301–4156 | e-ISSN 2460–5719



(a)

(b)

Gambar 6. Percobaan pengenalan *glyph*, (a) dalam kondisi normal dan (b) dalam kondisi miring.

A. KONFIGURASI PERANGKAT KERAS

Seperti yang ditunjukkan pada Gambar 4, INVys terdiri atas empat bagian perangkat keras utama, yaitu Microsoft Kinect 360 sebagai kamera RGB-D, baterai sebagai catu daya untuk Kinect, laptop/tablet komputer sebagai prosesor utama yang berisi perangkat lunak INVys, dan *earphone* sebagai alat untuk memberikan umpan balik kepada pengguna. Kinect menyediakan resolusi citra kedalaman 640 × 480, jangkauan maksimum 3.500 mm, serta sudut pandang 57,5° horizontal dan 43,5° vertikal [39], yang dianggap cocok untuk mengembangkan navigasi dalam ruangan berbasis penglihatan.

B. INDIKATOR PENGATURAN DAN KINERJA

1) *DETEKSI HAMBATAN*

Tujuan utama percobaan deteksi hambatan adalah untuk memastikan bahwa algoritma deteksi hambatan bekerja dengan baik dan untuk mengukur akurasi penghitungan jaraknya. Kamera RGB-D ditempatkan pada ketinggian 890 mm dari lantai (berdasarkan rata-rata tinggi pinggul orang Indonesia berusia 50 tahun [34]), seperti yang terlihat pada Gambar 5(a). Percobaan dilakukan dengan menempatkan enam objek yang berbeda yang biasa ditemukan di dalam ruangan (orang, kursi tipe 1, kursi tipe 2, sampah, kipas angin listrik, tumpukan kardus) pada 18 posisi yang berbeda (enam posisi untuk setiap area citra kedalaman), seperti yang terlihat pada Gambar 5(b) dan membandingkan jarak hambatan yang diperoleh sistem serta membandingkan jarak nyata dalam milimeter. Terdapat tiga metode yang diuji, yaitu pendekatan *baseline*, AAT, dan AADT. Selanjutnya, ketiga algoritma tersebut dibandingkan dengan menggunakan empat indikator kinerja.

Indikator pertama merupakan metrik kinerja yang paling penting, yaitu akurasi. Metrik ini ditunjukkan dengan perbedaan antara lokasi yang diperkirakan dan lokasi sebenarnya [40]. Hasil ini diperoleh dengan menghitung jarak rata-rata pengukuran kesalahan (μ_{obs}) menggunakan (8), dengan x adalah jarak sebenarnya halangan, x' adalah jarak yang dihitung oleh algoritma, dan n adalah jumlah percobaan.

$$\mu_{obs} = \sum \frac{|x - x'|}{n} \quad (8)$$

Indikator kedua adalah presisi. Jika akurasi hanya mengukur nilai kesalahan jarak rata-rata, presisi mengukur konsistensi sistem dalam bekerja. Indikator ini mengukur ketahanan algoritma karena dapat menunjukkan variasi kinerjanya setelah melalui banyak percobaan. Dalam teknik penentuan posisi, fungsi distribusi kumulatif *(cumulative distribution functions*, CDF) biasanya digunakan untuk mengukur presisi [40]. Penelitian ini juga menggunakan CDF untuk mengukur ketepatan algoritma yang dibandingkan.

Indikator ketiga adalah ketahanan akurasi sepanjang cakupan citra kedalaman. Jika presisi mengukur ketahanan terhadap jumlah percobaan, metrik ketiga ini mengukur ketahanan akurasi algoritma di sepanjang cakupan citra kedalaman. Metrik ini mempertimbangkan konsistensi keakuratan algoritma penghitungan jarak dari jarak terdekat hambatan hingga jarak hambatan mencapai batas cakupan kamera RGB-D. Hal ini penting karena akurasi piksel citra kedalaman Kinect menurun ketika jarak antara objek dan sensor meningkat [41]. Dengan demikian, pada dasarnya, akurasi algoritma penghitungan jarak dapat menurun ketika jarak objek yang diamati meningkat. Dari sudut pandang Kinect, hal ini disebabkan oleh adanya informasi kedalaman yang hilang pada citra kedalaman; sedangkan teknik penentuan posisi dengan ketahanan tinggi harus berfungsi dengan baik bahkan ketika beberapa sinyal tidak tersedia [40]. Oleh karena itu, indikator ini penting untuk mengevaluasi kinerja algoritma yang dibandingkan.

Indikator terakhir adalah waktu eksekusi. Indikator ini merupakan metrik untuk mengukur kecepatan algoritma bekerja dalam kondisi *real time*. Algoritma ini dijalankan pada *notebook* dengan prosesor Intel Core i3 1,8 GHz dan RAM 4 MB.

2) PENGENALAN *GLYPH*

Percobaan pengenalan *glyph* dilakukan dengan menggunakan lima pola *glyph* yang berbeda dan dengan dua ukuran yang berbeda, yaitu 6 × 6 cm (*glyph* A) dan 11,5 × 11,5 cm (*glyph* B). Hal ini dilakukan untuk mengetahui ukuran *glyph* terbaik yang dapat digunakan dalam konteks penyandang tunanetra, sehingga sistem dapat mengenali *glyph* pada jarak, yang dapat diterima, dari penyandang tunanetra. Percobaan ini mencoba mengenali *glyph* dengan jarak 300 mm dari kamera RGB-D hingga jarak maksimum ketika sistem tidak lagi dapat mengenali *glyph*. Untuk setiap jarak 300 mm, *glyph* diuji dalam dua skenario yang berbeda, yaitu kondisi normal dan miring.

Kondisi normal berarti *glyph* dikenali pada arah tegak lurus ke arah kamera RGB-D. Dalam skenario ini, *glyph* diuji pada empat posisi berbeda, yaitu posisi ideal (tidak diputar), diputar searah jarum jam 90°, diputar searah jarum jam 180°, dan diputar searah jarum jam 270°. Contoh percobaan ini dapat dilihat pada Gambar 6(a).

Dalam kondisi miring, sistem mencoba mengenali posisi *glyph* yang tidak tegak lurus terhadap kamera RGB-D. Percobaan ini dilakukan karena pada implementasinya, kamera RGB-D bergerak bersama dengan penyandang tunanetra, sehingga *glyph* tidak selalu terlihat tegak lurus terhadap kamera RGB-D. Pada skenario ini, *glyph* diuji pada empat posisi yang berbeda, yaitu *glyph* dimiringkan 45° ke belakang kiri, ke belakang kanan, ke belakang atas, dan ke belakang bawah. Contoh percobaan ini dapat dilihat pada Gambar 6(b).

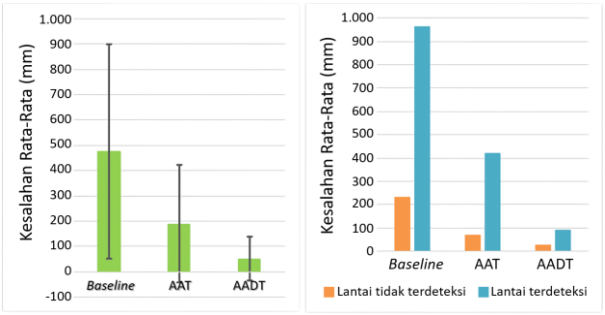
Dua indikator kinerja, yaitu akurasi dan waktu eksekusi, digunakan untuk mengevaluasi dua jenis *glyph* yang telah disebutkan sebelumnya. Akurasi membandingkan frekuensi sistem dapat mengenali *glyph* dalam beberapa kali uji coba untuk setiap rentang 300 mm. Nilainya diukur dalam persentase. Sementara itu, waktu eksekusi mengukur kecepatan sistem menjalankan algoritma, mulai dari akuisisi citra hingga pengenalan *glyph*.

C. HASIL

1) *DETEKSI HAMBATAN*

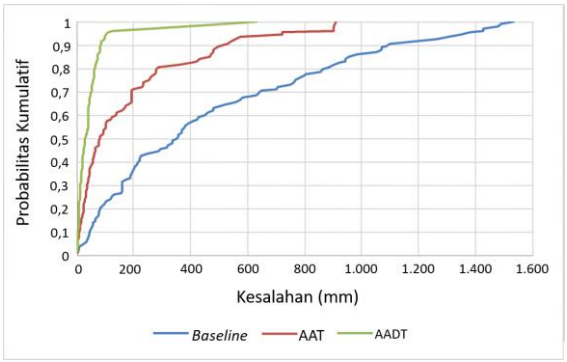
Gambar 7(a) menggambarkan kesalahan perhitungan jarak rata-rata dari setiap algoritma yang dibandingkan. Dapat dilihat bahwa AADT memiliki kesalahan jarak rata-rata terkecil

p-ISSN 2301–4156 | e-ISSN 2460–5719



(a) (b)

Gambar 7. Grafik kesalahan rata-rata, (a) kesalahan rata-rata perhitungan jarak untuk setiap algoritma pendeteksian rintangan, (b) perbandingan kesalahan jarak rata-rata untuk setiap algoritma ketika lantai terdeteksi dan tidak terdeteksi.



Gambar 8. Fungsi distribusi kumulatif (CDF) kesalahan penghitungan jarak pada banyak percobaan.

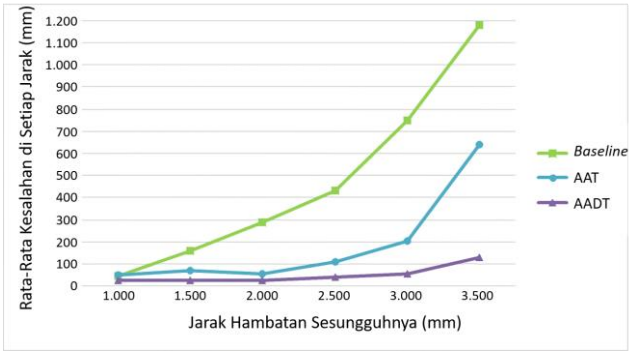
dibandingkan dengan algoritma yang lain. Kesalahan rata-rata AADT adalah 50,2 mm, dengan simpangan baku 87,7 mm. Selain perbandingan rata-rata kesalahan jarak di antara algoritma-algoritma tersebut, yang menarik untuk diamati juga adalah perbedaan rata-rata kesalahan jarak saat lantai terdeteksi (pada jarak lebih dari 2,5 m) dan tidak terdeteksi. Tujuannya untuk mengetahui keberhasilan AADT dalam mengatasi kelemahan AAT. Dapat dilihat pada Gambar 7(b) bahwa ketika lantai terdeteksi, semua algoritma yang dibandingkan menghasilkan perhitungan jarak yang lebih buruk daripada ketika lantai tidak terdeteksi. *Baseline* diambil dari penelitian sebelumnya [5]. Nilai ini dihitung berdasarkan kedalaman rata-rata di setiap blok. Nilai *baseline* akan memiliki kesalahan yang besar, terutama ketika nilai piksel terdekat berbeda.

AAT mengalami kesulitan ketika tidak dapat memisahkan objek dan lantai di depannya, terutama ketika lantai terdeteksi pada jarak lebih dari 2.500 mm. Pada kedua kondisi tersebut, kesalahan rata-rata AADT berada di bawah 100 mm. Hasil ini jauh lebih baik daripada pendekatan *baseline* dan AAT.

Dalam hal presisi, Gambar 8 menggambarkan CDF dari perhitungan kesalahan algoritma *baseline*, AAT, dan AADT. Berdasarkan Gambar 7(a), simpangan baku AADT adalah 87,7 mm, sehingga nilai maksimum perhitungan kesalahan yang dianggap normal berada pada kisaran 0 hingga 137,9 mm.

Meskipun demikian, dapat diamati pada Gambar 8 bahwa 93,5% kesalahan AADT berada di bawah 100 mm. Maka, dari 108 percobaan, sebagian besar percobaan menghasilkan kesalahan perhitungan jarak yang rendah. Bahkan AAT hanya memiliki ketepatan lokasi 53,7% dalam 100 mm (CDF kesalahan jarak 100 mm adalah 0,537). Jadi, AADT adalah algoritma yang memiliki presisi paling tinggi.

Keakuratan algoritma penghitungan jarak sering kali menurun jika jarak objek yang diamati bertambah. Artinya, nilai kesalahan dalam menghitung objek yang lebih dekat akan



Gambar 9. Ketangguhan akurasi algoritmea terhadap peningkatan jarak hambatan, di sepanjang cakupan kamera RGB-D.

TABEL I
WAKTU EKSEKUSI RATA-RATA DENGAN PENDEKATAN *BASELINE*, AAT, DAN AADT

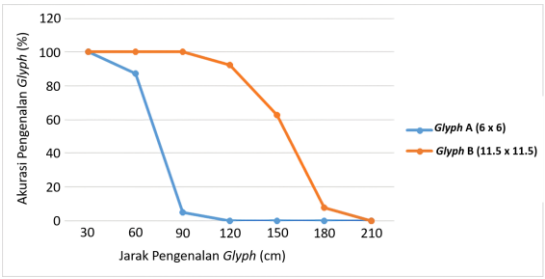
Algoritma	Waktu Eksekusi Rata-rata (ms)	Simpangan Baku
Baseline	7,41	1,19
AAT	7,05	1,22
AADT	9,50	3,61

lebih kecil dibandingkan objek yang lebih jauh. Gambar 9 menggambarkan ketangguhan algoritma yang dibandingkan dalam mengatasi permasalahan ini. Berdasarkan Gambar 9, akurasi *baseline* masih dapat diterima hingga jarak 2.000 mm; lebih dari jarak ini kesalahan jarak rata-rata terlalu tinggi. AAT bekerja dengan sangat baik pada jarak 2.500 mm atau kurang, dengan kesalahan jarak rata-rata di bawah 200 mm. Namun, jika jarak lebih dari 2.500 mm, kesalahan jarak rata-rata menjadi lebih tinggi, bahkan mencapai lebih dari 600 mm pada jarak 3.500 mm. Sementara itu, pada AADT dari jarak terdekat hingga batas cakupan kamera RGB-D, kesalahan jarak rata-rata secara konstan di bawah 150 mm. Kisaran kesalahan rata-rata AADT dari jarak pertama hingga jarak terakhir pengukuran hanya 103,3 mm. Oleh karena itu, akurasi penghitungan jarak AADT menurun sekitar 21,3 mm untuk setiap 500 mm peningkatan jarak hambatan dari kamera RGB-D.

Indikator kinerja terakhir adalah waktu eksekusi. Tabel I menunjukkan perbandingan waktu eksekusi antara pendekatan *baseline*, AAT, dan AADT. Pendekatan *baseline* dan AAT memiliki waktu eksekusi rata-rata yang sama, yaitu sekitar 7 ms dan dengan simpangan baku 1 ms hingga 1,5 ms. Sementara itu, AADT memiliki waktu eksekusi rata-rata yang lebih lambat, yaitu 9,5 ms, dan dengan simpangan baku 3,61. Jika dibandingkan dengan algoritma lainnya, hasil ini hanya menunjukkan sedikit perbedaan. Berdasarkan evaluasi kinerja yang telah dijelaskan, dapat disimpulkan bahwa AADT lebih baik daripada algoritma lainnya dalam hal akurasi, presisi, dan ketahanan terhadap peningkatan jarak rintangan. Namun, algoritma ini lebih lambat dalam hal waktu eksekusi, dengan perbedaan yang kecil.

2) PENGENALAN *GLYPH*

Gambar 10 menggambarkan perbandingan akurasi rata-rata pengenalan *glyph* antara *glyph* A (6 × 6 cm) dan *glyph* B (11,5 × 11,5 cm) untuk setiap jarak dalam uji coba. Gambar 10 menunjukkan bahwa *glyph* B masih memiliki akurasi rata-rata di atas 90% ketika jarak *glyph* dari kamera RGB-D mencapai 120 mm. Akurasi rata-rata *glyph* A pada jarak 120 mm adalah 0%, karena tidak ada *glyph* yang dapat dideteksi. Akurasi *glyph* A menurun secara signifikan pada jarak 600 mm atau lebih. Sebaliknya, pada jarak 150 mm atau lebih, akurasi *glyph* B



Gambar 10. Perbandingan rata-rata akurasi pengenalan *glyph* A (6 x 6 cm) dan *glyph* B (11,5 x 11,5 cm) untuk setiap jarak dalam uji coba.

TABEL II
WAKTU EKSEKUSI RATA-RATA PENGENALAN *GLYPH* A DAN *GLYPH* B

Informasi	*Glyph* A (6 × 6 cm)	*Glyph* B (11,5 × 11,5 cm)
Waktu eksekusi rata-rata (ms)	82	68
Waktu eksekusi tercepat (ms)	53	41
Waktu eksekusi terlambat (ms)	138	90

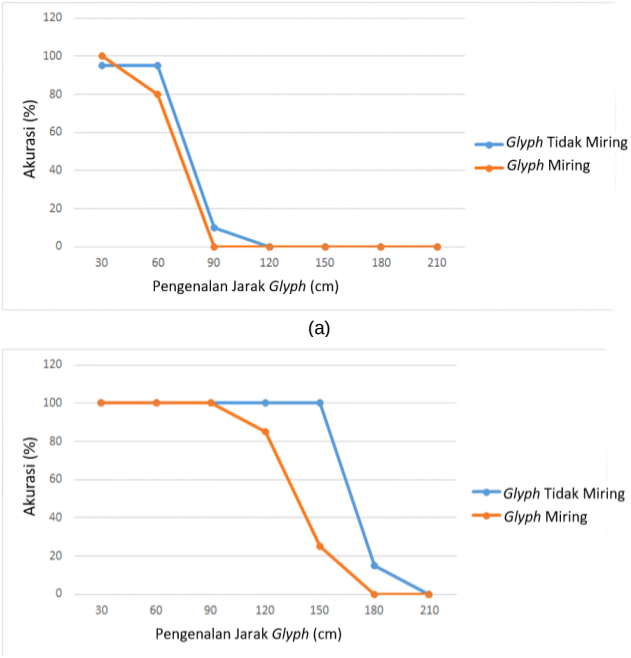
turun ke akurasi yang tidak dapat diterima bagi penyandang tunanetra.

Berdasarkan Gambar 10, dapat diambil kesimpulan sebagai berikut.

- Akurasi pengenalan *glyph* berbanding terbalik dengan jarak *glyph* dari kamera RGB-D. Artinya, akurasi menurun apabila jaraknya bertambah jauh.
- Ukuran *glyph* sebanding dengan akurasi pengenalan *glyph*. Jadi, makin besar ukuran *glyph*, makin baik akurasi pengenalan *glyph* untuk setiap jarak percobaan.
- *Glyph* B lebih baik digunakan daripada *glyph* A karena memiliki akurasi yang tinggi pada jarak 1.000 m hingga 1.500 mm. Oleh karena itu, INVys mengimplementasikan notifikasi suara pada jarak tersebut. Dengan demikian, penyandang tunanetra dapat mengenali *glyph* sebelum menabrak *glyph* tersebut.
- Berdasarkan poin ketiga, akurasi rata-rata *glyph* B dari jarak 300 mm hingga 1.500 mm adalah 91%. Hasil ini jauh lebih baik daripada *glyph* A yang hanya mampu mencapai akurasi rata-rata 38,5% pada jarak yang sama.
- Berdasarkan hasil keempat, ukuran *glyph* minimum yang praktis dan efektif dalam konteks navigasi di dalam ruangan bagi penyandang tunanetra adalah 11,5 × 11,5 cm atau lebih.

Tabel II merinci waktu eksekusi pengenalan rata-rata untuk *glyph* A dan *glyph* B. Waktu eksekusi *glyph* B lebih cepat daripada *glyph* A, meskipun kedua nilai tersebut kurang dari 100 ms. Oleh karena itu, *glyph* B tidak saja lebih mudah dikenali, tetapi juga lebih cepat. Namun, perbedaannya sangat kecil sehingga tidak signifikan.

Pengenalan *glyph* menganalisis akurasi jenis *glyph* pada kondisi miring dan tidak miring. Gambar 11 menunjukkan perbandingan akurasi pengenalan *glyph* A dan *glyph* B pada kondisi miring dan tidak miring. Gambar ini menunjukkan kondisi yang lebih berkontribusi terhadap penurunan akurasi pengenalan secara keseluruhan. Berdasarkan Gambar 11(a), pada sebagian besar jarak uji coba pengenalan *glyph*, *glyph* yang dimiringkan memiliki akurasi pengenalan yang lebih rendah daripada yang tidak dimiringkan. Hasil ini terlihat jelas pada Gambar 11(b). Pada jarak antara 1.200 mm dan 1.800 mm dari kamera RGB-D, *glyph* B yang dimiringkan memiliki akurasi yang lebih rendah daripada *glyph* B yang tidak dimiringkan, bahkan perbedaannya cukup signifikan. Pada jarak 1.500 mm, *glyph* yang tidak dimiringkan masih memiliki



Gambar 11. Perbandingan akurasi pengenalan *glyph* antara *glyph* miring dan *glyph* tidak miring pada (a) *glyph* A (6 x 6 cm) dan (b) *glyph* B (11,5 x 11,5 cm).

akurasi 100%, sedangkan *glyph* yang dimiringkan hanya memiliki akurasi 30%. *Glyph* yang dimiringkan memiliki dampak yang lebih signifikan dalam mengurangi akurasi pengenalan *glyph* secara keseluruhan daripada yang tidak dimiringkan.

VII. KESIMPULAN DAN PENELITIAN SELANJUTNYA

Pada makalah ini, INVys yang menggunakan kamera RGB-D telah dipresentasikan sebagai sistem navigasi dalam ruangan untuk penyandang tunanetra. INVys menggunakan kamera kedalaman untuk melakukan fungsi navigasi mikro dan menggunakan kamera berwarna untuk fungsi navigasi makro. Dalam navigasi mikro, INVys mengusulkan algoritma AADT untuk mendeteksi hambatan.

Hasil percobaan menunjukkan bahwa AADT lebih unggul dari algoritma lain dalam hal akurasi (kesalahan jarak 50,2 mm), presisi (95% kesalahan dalam uji coba berada di bawah 100 mm), dan ketahanan terhadap peningkatan jarak hambatan (kesalahan jarak rata-rata secara konstan di bawah 150 mm dari jarak yang lebih dekat ke batas cakupan kamera RGB-D). Satu-satunya indikator kinerja yang lebih buruk dari algoritma lain adalah waktu eksekusi. Namun, perbedaannya sangat kecil (sekitar 3 ms) sehingga tidak signifikan.

Dalam hal fungsi navigasi makro, INVys mengenali *optical glyph* sebagai penanda titik-titik penting atau rambu-rambu navigasi yang ditempatkan di dalam ruangan dengan menggunakan metode *automatic glyph binarization*. Uji coba dengan dua ukuran *glyph* yang berbeda, yaitu A (6 × 6 cm) dan B (11,5 × 11,5 cm), menunjukkan bahwa *glyph* B memiliki akurasi rata-rata 91% pada jarak 300 mm hingga 1.500 mm. Hasil ini jauh lebih baik daripada *glyph* A yang hanya memiliki akurasi rata-rata 38,5% pada jarak yang sama. Untuk penelitian di masa mendatang, akan sangat bagus jika dapat ditambahkan sistem pemosisian dan fungsi perencanaan jalur yang tidak bergantung pada penanda apa pun. Hal ini dapat digunakan untuk memandu para penyandang tunanetra berjalan selangkah demi selangkah dari satu tempat ke tempat lain tanpa hambatan.

KONFLIK KEPENTINGAN

Penulis menyatakan bahwa tidak terdapat konflik kepentingan.

KONTRIBUSI PENULIS

Konseptualisasi, Widyawan dan Muhammad Risqi Saputra; metodologi, Widyawan dan Paulus Insap Santosa; perangkat lunak, Muhammad Risqi Saputra; validasi, Widyawan; analisis formal, Widyawan dan Paulus Insap Santosa; investigasi, Widyawan dan Muhammad Risqi Saputra; sumber daya, Muhammad Risqi Saputra; kurasi data, Paulus Insap Santosa; penulisan—penyusunan draf asli, Widyawan dan Muhammad Risqi Saputra; penulisan—peninjauan dan penyuntingan, Paulus Insap Saputra; supervisi, Widyawan dan Paulus Insap Saputra.

REFERENSI

[1] Y. Wu, H.B. Zhu, Q.X. Du, dan S.M. Tang, "A Survey of the Research Status of Pedestrian Dead Reckoning Systems Based on Inertial Sensors," *Int. J. Automat., Comput.*, Vol. 16, No. 1, hal. 65–83, Feb. 2019, doi: 10.1007/s11633-018-1150-y.

[2] H. Petrie, "User Requirements for a GPS-Based Travel Aid for Blind People," dalam *Proc. Conf. Orientat., Navig. Syst. Blind Persons*, 1995.

[3] W.C.S.S. Simões, dkk., "A Review of Technologies dan Techniques for Indoor Navigation Systems for the Visually Impaired," *Sens.*, Vol. 20, No. 14, hal. 1–35, Jul. 2020, doi: 10.3390/s20143935.

[4] H. Takizawa, Y. Kuramochi, dan M. Aoyagi, "Kinect Cane System: Recognition Aid of Available Seats for the Visually Impaired," *2019 IEEE 1st Glob. Conf. Life Sci., Technol. (LifeTech)*, 2019, hal. 189–193, doi: 10.1109/LifeTech.2019.8884061.

[5] A. Khan, F. Moideen, dan J. Lopez, "KinDectect: Kinect Detecting Objects," dalam *Computers Helping People with Special Needs*, K. Miesenberger dkk., Eds., Heidelberg, Jerman: Springer, 2012, hal. 588–595, doi: 10.1007/978-3-642-31534-3_86.

[6] B. Singh, "A Framework of Connected Smart Sensing Canes for Obstacle Detection and Avoidance," *2022 IEEE 10th Region 10 Humanit. Technol. Conf. (R10-HTC)*, 2022, hal. 76–80, doi: 10.1109/r10-htc54060.2022.9930047.

[7] M.A. Ikbal, F. Rahman, dan M.H. Kabir, "Microcontroller Based Smart Walking Stick for Persons with Visual Impairment," *2018 4th Int. Conf. Elect. Eng., Inf., Commun. Technol. (iCEEiCT)*, 2018, hal. 255–259, doi: 10.1109/CEEICT.2018.8628048.

[8] H. Watanabe, M. Sumiya, dan T. Terada, "Human-Machine Cooperative Echolocation Using Ultrasound," *IEEE Access*, Vol. 10, hal. 125264–125278, 2022, doi: 10.1109/ACCESS.2022.3224468.

[9] M.H.A. Wahab dkk., "Smart Cane: Assistive Cane for Visually-Impaired People," *Int. J. Comput. Sci*, Vol. 8, No. 4, hal. 21–27, Jul. 2011.

[10] Y. Zhao dkk., "Laser Based Navigation in Asymmetry and Complex Environment," *Symmetry*, Vol. 14, No. 2, hal. 1-18, Jan. 2022, doi: 10.3390/sym14020253.

[11] S. Real dan A. Araujo, "Navigation Systems for the Blind and Visually Impaired: Past Work, Challenges, and Open Problems," *Sens.*, Vol. 19, No. 15, hal. 1–20, Agu. 2019, doi: 10.3390/s19153404.

[12] N.A. Kumar, Y.H. Thangal, dan K.S. Beevi, "IoT Enabled Navigation System for Blind," *2019 IEEE R10 Humanit. Technol.Conf. (R10-HTC)(47129)*, 2019, hal. 186–189, doi: 10.1109/R10-HTC47129.2019.9042483.

[13] P.S. Farahsari, A. Farahzadi, J. Rezazadeh, dan A. Bagheri, "A Survey on Indoor Positioning Systems for IoT-Based Applications," *IEEE Internet Things J.*, Vol. 9, No. 10, hal. 7680–7699, Mei 2022, doi: 10.1109/JIOT.2022.3149048.

[14] D.R. Bolla, S. Trisheela, P. Nagarathana, dan H. Sarojadevi, "Object Detection in Computer Vision Using Machine Learning Algorithm for Persons with Visual Impairment," *2022 IEEE 2nd Mysore Sub Sect. Int. Conf. (MysuruCon)*, 2022, hal. 1–6, doi: 10.1109/MysuruCon55714.2022.9972494.

[15] R. Oktem dan E. Aydin, "An RFID Based Indoor Tracking Method for Navigating Impaired People," *Turkish J. Elect. Eng., Comput. Sci.*, Vol. 18, No. 2, hal. 185–196, Mar. 2010, doi: 10.3906/elk-0904-3.

[16] S. Koley dan R. Mishra, "Voice Operated Outdoor Navigation System for Visually Impaired Persons," *Int. J. Eng. Trends, Technol.*, Vol. 3, No. 2, hal. 153–157, Mar.–Apr. 2012.

[17] D. Ni dkk., "The Design and Implementation of a Walking Assistant System with Vibrotactile Indication dan Voice Prompt for the Visually Impaired," *2013 IEEE Int. Conf. Robot., Biomimetics (ROBIO)*, 2013, hal. 2721–2726, doi: doi: 10.1109/ROBIO.2013.6739885.

[18] M.R.U. Saputra, Widyawan, G.D. Putra, dan P.I. Santosa, "Indoor Human Tracking Application Using Multiple Depth-Cameras," *2012 Int. Conf. Adv. Comput. Sci., Inf. Sys. (ICACSIS)*, 2012, hal. 307–312.

[19] A. Kirillov (2011) "From Glyph Recognition to Augmented Reality," [Online], https://www.codeproject.com/Articles/258856/From-glyph-recognition-to-augmented-reality, tanggal akses: 12-Okt-2022.

[20] D.R. Bolgiano dan E.D. Meeks, "A Laser Cane for The Blind," *IEEE J. Quantum Electron.*, Vol. 3, No. 6, hal. 268–268, Jun. 1967, doi: 10.1109/JQE.1967.1074528.

[21] D.I. Ahlmark, H. Fredriksson, dan K. Hyyppä, "Obstacle Avoidance Using Haptics and a Laser Rangefinder," *IEEE Workshop Adv. Robot., Its Soc. Impacts (ARSO)*, 2013, hal. 76–81, doi: 10.1109/ARSO.2013.6705509.

[22] T. Hiramoto, T. Araki, dan T. Suzuki, "Infrared Guided White Cane for Assisting the Visually Impaired to Walk Alone," *2022 Int. Conf. Mach. Learn., Cybern. (ICMLC)*, 2022, hal. 276–280, doi: 10.1109/icmlc56445.2022.9941336.

[23] E.B. Kaiser dan M. Lawo, "Wearable Navigation System for the Visually Impaired and Blind People," *2012 IEEE/ACIS 11th Int. Conf. Comput. Inf. Sci.*, 2012, hal. 230–233, doi: 10.1109/ICIS.2012.118.

[24] J.M. Benjamin, "The Laser Cane," *Bull. Prosthet. Res.*, hal. 443–450, 1974.

[25] L.A. Guerrero, F. Vasquez, dan S.F. Ochoa, "An Indoor Navigation System for the Visually Impaired," *Sens.*, Vol. 12, No. 6, hal. 8236–58, Jun. 2012, doi: 10.3390/s120608236.

[26] A. Riener dan H. Hartl, "'Personal Radar': A Self-governed Support System to Enhance Environmental Perception," *Proc. 26th Annu. BCS Interact. Spec. Group Conf. People, Comput.*, 2012, hal. 147–156.

[27] Y. Wei dan M. Lee, "A Guide-Dog Robot System Research for the Visually Impaired," *IEEE Int. Conf. Ind. Technol. (ICIT)*, 2014, hal. 800–805, doi: 10.1109/ICIT.2014.6894906.

[28] A. Aladren, G. Lopez-Nicolas, L. Puig, dan J.J. Guerrero, "Navigation Assistance for the Visually Impaired Using RGB-D Sensor with Range Expansion," *IEEE Syst. J.*, Vol. 10, No. 3, hal. 922–932, Sep. 2016, doi: 10.1109/JSYST.2014.2320639.

[29] D. Dakopoulos dan N.G. Bourbakis, "Wearable Obstacle Avoidance Electronic Travel Aids for Blind: A Survey," *IEEE Trans. Syst., Man, Cybern. Part C (Appl., Rev.)*, Vol. 40, No. 1, hal. 25–35, Jan. 2010, doi: 10.1109/TSMCC.2009.2021255.

[30] D. Bernabei dkk., "A Low-Cost Time-Critical Obstacle Avoidance System for the Visually Impaired," *2011 Int. Conf. Indoor Position., Indoor Navig. (IPIN)*, 2011, hal. 21–23.

[31] V. Mohandas dan R. Paily, "Stereo Disparity Estimation Algorithm for Blind Assisting System," *CSI Trans. ICT*, Vol. 1, No. 1, hal. 3–8, Mar. 2013, doi: 10.1007/s40012-012-0004-y.

[32] R.G. Praveen dan R.P. Paily, "Blind Navigation Assistance for Visually Impaired Based on Local Depth Hypothesis from a Single Image," *Procedia Eng.*, Vol. 64, hal. 351–360, 2013, doi: 10.1016/j.proeng.2013.09.107.

[33] C. Lee, Y. Su, dan L. Chen, "An Intelligent Depth-Based Obstacle Detection System for Visually-Impaired Aid Applications," *2012 13th Int. Workshop Image Anal. Multimed. Interact. Serv.*, Mei 2012, hal. 1–4, doi: 10.1109/WIAMIS.2012.6226753.

[34] T.K. Chuan, M. Hartono, dan N. Kumar, "Anthropometry of the Singaporean and Indonesian Populations," *Int. J. Ind. Ergonom.*, Vol. 40, No. 6, hal. 757–766, Nov. 2010, doi: 10.1016/j.ergon.2010.05.001.

[35] M.R.U. Saputra, Widyawan, dan P.I. Santosa, "Obstacle Avoidance for Visually Impaired Using Auto-Adaptive Thresholding on Kinect's Depth Image," *2014 IEEE 11th Intl. Conf. Ubiquitous Intell., Comput., 2014 IEEE 11th Intl. Conf. Autonomic, Trusted Comput., 2014 IEEE 14th Intl. Conf. Scalable Comput., Commun., Its Associated Workshops*, 2014, hal. 337–342, doi: 10.1109/UIC-ATC-ScalCom.2014.108.

[36] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Trans. Syst. Man, Cybern.*, Vol. 9, No. 1, hal. 62–66, Jan. 1979, doi: 10.1109/TSMC.1979.4310076.

[37] L. Chen, X. Nguyen, dan C. Liang, "Object Segmentation Method Using Depth Slicing and Region Growing Algorithms," *Int. Conf. 3D Syst., Appl. Gen.*, 2010, hal. 4–7.

[38] F.Y. Shih, *Image Processing and Pattern Recognition: Fundamentals and Techniques*, 4th ed. Hoboken, AS: Wiley-IEEE Press, 2010.

[39] *Kinect for Windows, Human Interface Guidelines v1.7*, Microsoft, Redmond, WA, USA, 2013.

[40] X. Guo dkk., "A Survey on Fusion-Based Indoor Positioning," *IEEE Commun. Surv., Tut.*, Vol. 22, No. 1, hal. 566–594, 2020, doi: 10.1109/COMST.2019.2951036.

[41] J. Han, L. Shao, D. Xu, dan J. Shotton, "Enhanced Computer Vision with Microsoft Kinect Sensor: A Review," *IEEE Trans. Cybern.*, Vol. 43, No. 5, hal. 1318–1334, Okt. 2013, doi: 10.1109/TCYB.2013.2265378.