Pengembangan Algoritma Pengenalan Bentuk dan Arah Objek pada Sistem *Omnidirectional Vision Sensor*

Bagus Arthaya[a], Ali Sadiyoko[b], Chandra Wiejaya
Jurusan Teknik Industri, Fakultas Teknologi Industri, Universitas Katolik Parahyangan, Bandung
Email: [a]bagusart@home.unpar.ac.id, bagusmooi@gmail.com, [b]alfa51@home.unpar.ac.id

ABSTRAK

Otomasi merupakan salah satu solusi untuk meningkatkan efisiensi dan keamanan dalam dunia produksi. *Autonomous Mobile Robot* (AMR) merupakan salah satu bentuk dari teknologi otomasi. AMR memiliki kemampuan navigasi sehingga dapat bergerak pada lingkungan yang tidak dikenalnya. Kemampuan ini dapat digunakan untuk menjelajahi daerah-daerah yang tidak dikenali dan berbahaya bagi manusia, misalnya daerah sumber gas yang baru ditemukan. AMR sebagai salah satu bentuk teknologi otomasi membutuhkan komponen sensor untuk dapat mengenali lingkungan sekitarnya. Salah satu jenis sensor yang dapat digunakan pada teknologi otomasi adalah sensor kamera. Sensor kamera memiliki karakteristik yang sama dengan indera penglihatan (mata) pada makhluk hidup.Kelemahan sensor kamera adalah keterbatasan sudut pandangnya. Sudut pandang yang dapat ditangkap oleh sensor kamera serupa dengan sudut pandang pada mata manusia (ke arah kamera menghadap/ke arah depan), padahal AMR membutuhkan data mengenai lingkungan yang berada di bagian belakang sensor kamera. Hal ini dapat diatasi dengan menggunakan sistem omnidirectional vision sensor. Sistem tersebut dapat memberikan informasi mengenai 360^0 lingkungan sekitar hanya dalam satu image. Sistem ini didasarkan pada karakteristik mata serangga. Sistem ini dapat dibuat dengan mengarahkan kamera ke sebuah cermin cembung. Semakin cembung cermin yang digunakan, maka sudut pandang yang dihasilkan akan semakin luas. Sebelum menentukan gerakan dari AMR, pertama-tama AMR harus dapat mendeteksi adanya objek di sekitarnya, sehingga tabrakan dapat dihindari. Penelitian ini memfokuskan penelitian pada pengenalan bentuk objek dan penentuan arah objek tersebut ditinjau dari sistem AMR yang menggunakan omnidirectional vision sensor. Sistem omnidirectional vision sensor yang dibuat hanya digunakan sebagai sarana pengambilan data, dan belum diintegrasikan dengan AMR.

Kata kunci: AMR, omnidirectional vision, penentuan arah, pengenalan bentuk.

ABSTRACT

One of application of automation technology is Autonomous Mobile Robot (AMR), which has a navigation so that it can move around the unknown environment. This AMR ability can be used for moving around the dangerous area for people such as a place where new gas is found. As an application of automation technology, AMR needs a sensor for recognizing the environement. One of the sensors used in the automation technology and application is camera, which has the same characteristic as human eyes. Drawback of camera sensor is a limitation in the field of view of the camera. Camera can only catch the image or data of an object facing to the camera, whereas AMR neesd to know the data in the other direction or field of view. This can be overcomed by using omnidirectional vision sensor system. This system can give information of environment around the camera (360^0) in one image. This system can be done by facing the camera to a convex mirror. Fisrtly, AMR has to detect the existence of the objects around it so that the collision can be avoided. This research focused on regocnizing an object and determining the direction of that object related to AMR. The designed omnidirectional vision sensor system is only used as the tool for data acquisition and has not been integrated with AMR.

***Keywords**: AMR, omnidirectional vision, direction determination, recognition*

PENDAHULUAN

Otomasi dapat meningkatkan efisiensi dan keselamatan kerja dalam dunia produksi. Salah satu aplikasi otomasi adalah AMR (*Autonomous Mobile Robot*). AMR membutuhkan komponen sensor untuk mengenali lingkungan sekitar. Salah satu jenis sensor yang dapat digunakan adalah sensor kamera. Kelemahan sensor kamera adalah keterbatasan sudut pandang yang dapat dilihat pada satu *image* yang ditangkapnya. Oleh karenanya sistem kamera ini relatif agak sulit diterapkan pada sistem AMR sederhana. Namun, kelemahan ini dapat diatasi dengan menggunakan *onnidirectional vision sensor* [1]. Sistem tersebut dapat diperoleh dengan mengarahkan sebuah sensor kamera menuju ke arah cermin berbentuk setengah bola yang diletakkan di atasnya.

Penggunaan *omnidirectional image sensor* pada proses navigasi AMR memerlukan pengolahan data *image* lebih lanjut untuk digunakan dalam menentukan arah gerak dan menentukan posisi dan

Catatan: Diskusi untuk makalah ini diterima sebelum tanggal 1 Desember 2007. Diskusi yang layak muat akan diterbitkan pada Jurnal Teknik Elektro volume 8, nomor 1, Maret 2008.

bentuk objek yang akan ditangani saat AMR berfungsi sebagai *material handling* ataupun dapat menjadi halangan yang harus dihindari. Untuk lebih memfokuskan arah penelitian ini, maka rumusan masalah dapat dinyatakan sebagai berikut:

a. Bagaimana mengembangkan algoritma yang diperlukan untuk mengolah data *image* yang dihasilkan oleh sistem *omnidirectional image sensor*?
b. Bagaimana mengembangkan algoritma untuk menentukan bentuk dan posisi objek yang ditangkap oleh sistem sensor di atas?

METODA PENELITIAN

Penelitian ini diawali dengan penelaahan terhadap konsep sistem *omnidirectional image sensor* dan model data yang dihasilkan dari image yang ditangkap oleh suatu sistem sensor kamera. Selanjutnya dikembangkan algoritma yang diperlukan untuk mengolah data *image* tersebut agar didapatkan output berupa bentuk dan arah objek terhadap sistem sensor tersebut.

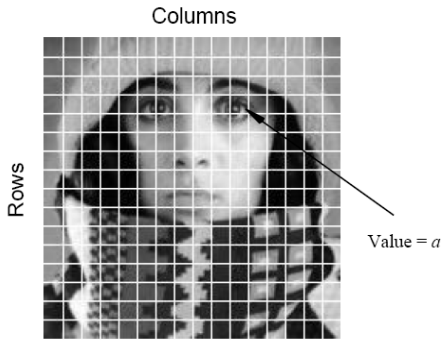
Suatu model sederhana dari sistem *omnidirectional image sensor* kemudian disiapkan dan bentuk-bentuk benda sederhana ditampilkan dalam sistem ini. Bentuk awal yang digunakan adalah lingkaran, persegi panjang dan segitiga. Objek tersebut diatur agar memiliki tingkat kekontrasan yang tinggi (berwarna hitam dan putih) dan diletakkan secara manual di dalam area tangkapan sensor. Pengaruh dari variasi tingkat pencahayaan yang ada dalam ruangan sementara diabaikan.

Image yang ditangkap oleh kamera diproses dan diolah lebih lanjut dalam perangkat lunak MATLAB dengan memperhatikan proses terbentukanya gambar benda pada sistem *omnidirectional image sensor*. Langkah-langkah yang terkait dengan pengolahan image, seperti pembacaan *image*, *cropping*, konversi ke biner, penentuan jumlah objek, pembuatan pandangan panorama, penentuan tepi objek dan lain-lain. Dari beberapa bentuk benda yang ditampilkan, dapat diambil kesimpulan yang terkait dengan tujuan penelitian ini.

TEORI DASAR

Image

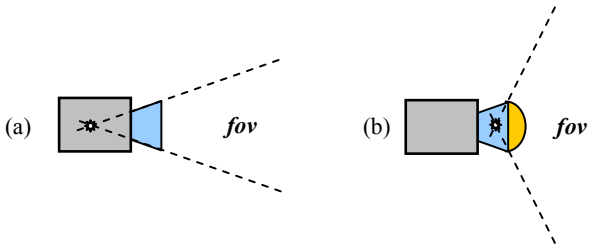
Image merupakan suatu data digital yang dihasilkan oleh sistem sensor berupa kamera. Dalam pengolahan data, *image* direpresentasikan sebagai matriks. Matriks representasi tersebut memiliki jumlah baris dan kolom sesuai dengan resolusi *image* yang diwakilinya. Perpotongan antara baris dan kolom disebut pixel, dan berisi nilai entri matriks yang sesuai dengan warna pixel yang sama pada *image* yang diwakilinya. Misalnya untuk *image* 8 bit, nilai entri matriks berada pada range 0-255. Nilai entri tersebut memiliki hubungan dengan warna yang ada pada *image* yang diwakilinya, misalnya nilai entri 0 untuk hitam dan 1 untuk putih.



Gambar 1. Representasi *Image* Sebagai Matriks [2]

Omnidirectional Image dan *Panoramic View*

Kamera konvensional memiliki keterbatasan pandangan seperti ditunjukkan dalam Gambar 2a. Untuk meningkatkan daerah tangkapan gambar, Cao et. al. [3] menggunakan sistem yang menggunakan *fish eye lens*. Sistem ini menggunakan lensa dengan ketebalan yang sangat besar untuk menggantikan lensa pada kamera konvensional. Lensa khusus tersebut dapat memberikan *field of view* (*fov*) yang lebih besar daripada kamera konvensional. Namun, daerah pandang yang dihasilkan masih belum dapat memenuhi kebutuhan data seluruh lingkungan sekitar (Gambar 2b) khususnya untuk mengendaliah AMR.



Gambar 2. Perbandingan Sudut Pandang Kamera Biasa dengan *Fish Eye Lens* [4]

Bergabagai usaha telah ditempuh oleh banyak peneliti untuk mendapatkan daerah pandang kamera yang luas. Bruckstein dan Richardson [5] membuat desain yang menggunakan 2 *parabolic mirror*, 1 *convex*, dan 1 *concave*. Nayar [4] membuat desain menggunakan 2 cermin dengan bentuk konis. Sementara Nalwa [6] menggunakan susunan penempatan 4 *triangular planar mirrors* yang saling bersisian dalam bentuk susunan piramid dengan setiap kamera di bawah setiap cermin. Pada tahun 1843 [7], Joseph Puchberger dari Retz Austria menemukan sistem yang menggunakan kamera yang berputar untuk

mendapatkan *omnidirectional image*. Dengan menggunakan sistem *omnidirectional vision sensor* akan diperoleh sudut pandang yang lebih luas dari kedua sistem sebelumnya (Gambar 3a). Hanya saja *image* yang diperoleh dengan menggunakan sistem ini akan mengalami distorsi [8] karena sifat dasarnya yang menyajikan seluruh gambar lingkungan dalam satu bidang berbetuk lingkaran. Dengan mengetahui sifat-sifat dari *omnidirectional image* yang dihasilkan, maka distorsi ini dapat diperbaiki sehingga *image* yang diperoleh dapat diolah dengan lebih mudah. Salah satu jenis perbaikan distrosi yang dilakukan terhadap *omnidirectional image* adalah dengan mengubah *image* tersebut menjadi *panoramic view* (Gambar 3b)..

Jenis *image* ini memberikan sudut pandang 360^O dari lingkungan sekitar sensor dan cocok untuk digunakan dalam melakukan *tracking* garis-garis lurus, terutama garis vertikal. Seluruh titik 3D yang terletak pada sudut elevasi yang sama pada keadaan nyata akan ditampilkan menjadi sebuah lingkaran 2D pada *omnidirectional image*. Dengan mengetahui sifat ini, maka dapat disimpulkan bahwa perubahan dari *omnidirectional image* menjadi *panoramic view* merupakan perubahan dari koordinat Kartesian menjadi koordinat Polar. Korelasi antara informasi dari sistem *panoramic view* dengan sistem *omnidirectional* dapat dinyatakan dalam Pers. 1 berikut:

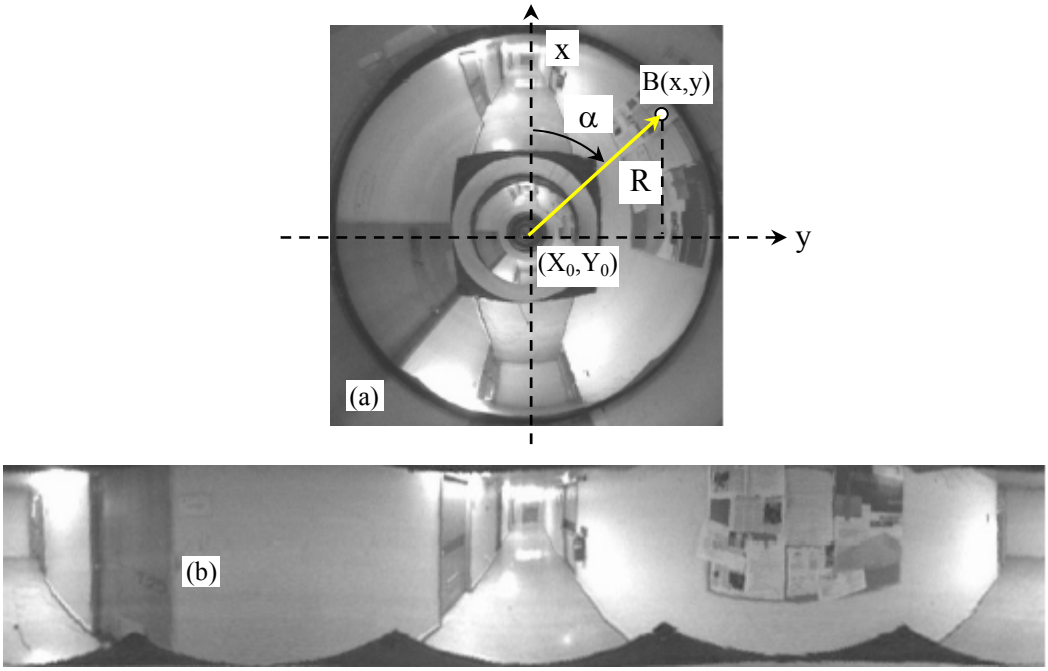
$$Panoramic(\alpha, R) = Omni(R\cos\alpha + Xo, R\sin\alpha + Yo) \quad (1)$$

di mana α adalah koordinat sudut, R adalah koordinat jarak radial, dan (X_0,Y_0) adalah koordinat pusat dari *omnidirectional image*. Jadi, *panoramic view* memiliki sumbu koordinat sudut (sumbu X) terhadap jarak radial (sumbu Y). Pemilihan besarnya nilai sudut (α) dan jarak radial (R) disesuaikan dengan resolusi dari *omnidirectional image* yang dihasilkan.

PEMBUATAN SISTEM *OMNIDIRECTIONAL VISION SENSOR*

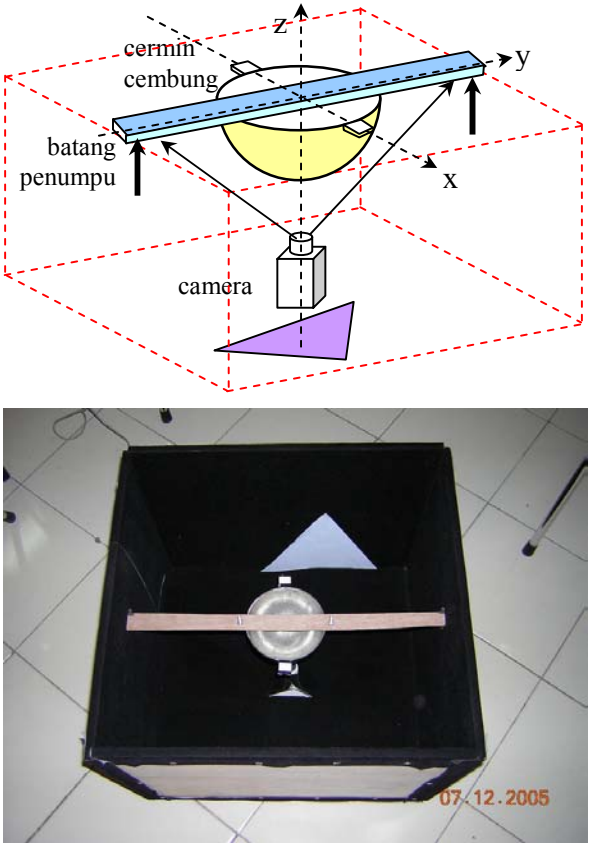
Penelitian ini difokuskan pada algoritma pengenalan bentuk dan arah objek dengan menggunakan pembatasan masalah berupa objek dan lingkungan memiliki perbedaan intensitas yang kontras. Hal ini menyebabkan sistem *omnidirectional vision sensor* yang dibuat belum dapat langsung diterapkan pada AMR yang berjalan pada lingkungan yang dinamis. Sistem yang dibuat hanya digunakan sebagai sarana penelitian awal dengan pengambilan data *image* pada lingkungan yang masih sangat terbatas.

Sistem *omnidirectional* sederhana dibuat menggunakan sebuah kotak terbuka di satu sisi yang bagian dalamnya dilapisi dengan kain beludru hitam (lingkungan yang berwarna hitam). Objek yang kontras didapat dengan menggunakan kertas berwarna putih yang ditempelkan pada area hitam di dalam kotak baik pada lantai ataupun dinding kotak. Cermin *spherical* dibuat dari plat logam aluminium berbentuk setengah bola berdiameter 12 cm yang dilapisi dengan krom pada bagian luarnya dan dipasang pada bagian tengah atas kamera. Tampilan 3D model sistem *omnidirectional vision sensor* ini ditunjukkan pada Gambar 4. Cermin cembung dipasang pada suatu batang penumpu yang disangga oleh dinding ruang kotak hitam. Kamera (sensor) ditempatkan



Gambar 3. Contoh Sebuah *Omnidirectional Image* (a) dan *Panoramic View*nya (b) **[8]**.

tepat di bagian tengah bawah kotak dan menghadap ke arah sumbu tengah cermin sehingga kamera dapat menangkap gambar seluruh bagian cermin dan batang penumpu. Selanjutnya arah sumbu-x akan dijadikan acuan dalam algortima untuk menentukan arah (sudut) di mana objek berada.



Gambar 4. *Omnidirectional Vision Sensor* yang Digunakan dalam Penelitian

PENGEMBANGAN ALGORITMA PENGOLAHAN DATA *IMAGE*

Pembacaan Data *Image*

Tahap pertama yang dilakukan adalah membaca data *image* yang telah diperoleh dari sistem yang telah dibuat. Tahapan ini akan menghasilkan matriks representasi dari data *image* yang dapat diolah secara matematis. Data *image* yang dibaca dan matriks representasinya memiliki hubungan timbal balik. Proses pembacaan data *image* pada MATLAB dilakukan dengan menggunakan perintah *imread*. Input dari perintah ini adalah *omnidirectional image* yang akan diolah. *Omnidirectional image* yang akan diolah ini memiliki jenis gambar *grayscale* dengan kedalaman 8 bit (dengan nilai entri yang berada dalam range 0-255) dan resolusi sebesar 320 x 240. Perintah yang digunakan untuk membaca *image* adalah:

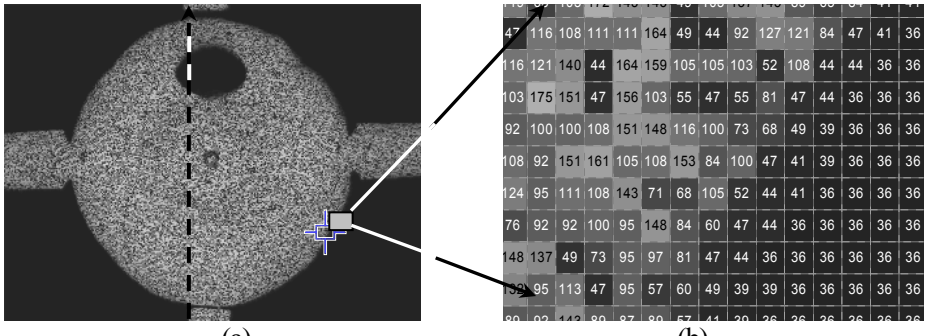
image = imread ('Foto1.bmp');

Satu gambar yang ditangkap dari kamera akan digunakan sebagai contoh aplikasi algoritma pada pengolahan data (Gambar 5a). Bila dilihat lebih mendalam, suatu bagian pembesaran dari gambar ini akan menunjukkan nilai-nilai biner dari setiap pixel yang membentuk gambar tersebut (Gambar 5b). Terlihat bahwa garis tepi gambar yang tertangkap kamera tidak begitu tegas, sehingga perlu proses lanjut terutama pada langkah penentuan bentuk objek yang tertangkap oleh kamera.

Image Cropping

Tahapan selanjutnya yang dilakukan adalah melakukan *cropping* (pemotongan) terhadap *image* hasil pembacaan data. Tujuan dari proses ini adalah untuk menghilangkan data yang tidak terkait langsung dengan objek penelitian dan sekaligus memperkecil dimensi matriks, sehingga proses pengolahan selanjutnya akan memakan waktu yang lebih singkat.

Sebagai contoh bagian batang penyangga di sebelah kiri dan kanan cermin cembung (pada Gambar 5a) masih tertangkap oleh kamera berupa batang horizontal. Bagian gambar ini sebaiknya dihilangkan sehingga yang perlu diolah hanyalah bentuk lingkaran yang merupakan *image* dari cermin cembung dan semua objek yang dipantulkan oleh cermin ini. Pemotongan dilakukan dengan memanggil fungsi **imcrop** pada MATLAB. Contoh penggunaan fungsi ini adalah sebagai berikut:



(a) (b)

Gambar 5. Data *image* (a) dan Sebagian Matriks Representasi *Image* Tersebut (b)

crop = imcrop (image, [49 4 224 230]);

Hasil langkah ini adalah suatu image dengan ukuran (224-49) x (230-4) pixel. Nilai batas pemotongan ini dibuat tetap karena posisi kamera terhadap cermin selalu tetap sehingga luas bidang gambar yang ditangkap kamera akan selalu sama dari waktu ke waktu.

Konversi *Cropped Image* Menjadi *Binary Image*

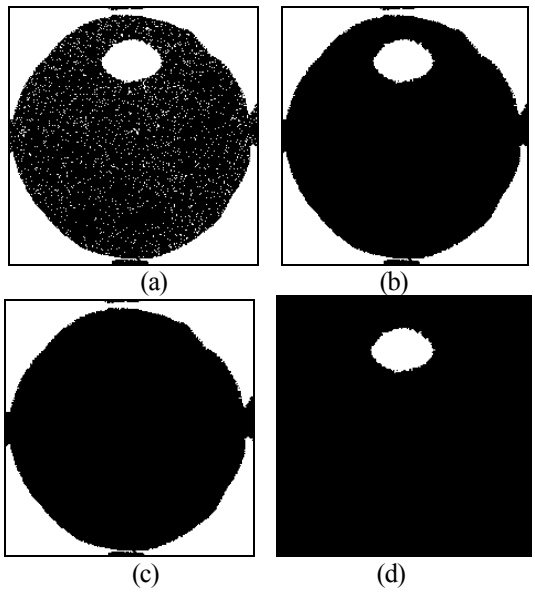
Tahapan ini bertujuan untuk mengubah *image* hasil *cropping* menjadi *image* dalam bentuk bilangan biner. Langkah ini dilakukan dengan menggunakan nilai *threshold* yang akan menjadi garis batas pengubahan suatu intensitas cahaya menjadi warna hitam atau putih. Nilai entri matriks (yang mewakili suatu intensitas cahaya) akan diubah menjadi warna hitam apabila berada di bawah nilai *threshold*, dan begitu sebaliknya yaitu diubah menjadi warna putih apabila berada di atas nilai *threshold* tadi. Fungsi MATLAB untuk melakukan konversi dimulai dengan menentukan level nilai *threshold* (menggunakan metode Otsu [9]) dengan perintah *graythresh* sebagai berikut:

level = graythresh(crop);

Output dari perintah ini adalah suatu nilai yang berada dalam range 0-1. Binary image didapat dengan menerapkan perintah *~im2bw* terhadap input bernama *crop*, dan perintahnya adalah:

binary = ~im2bw(crop,level-0.17);

Angka 0,17 didapat dengan cara *trial and error*, dimana angka ini memnberikan hasil yang terbaik. Gambar 6a menunjukkan hasil dari proses konversi tersebut.



Gambar 6. *Image* Hasil Cropping (a) dan *Image* Setelah Noise Dihilangkan (b, c) Serta Objek yang Berhasil Diisolasi dari Gambar Keseluruhan

Penghilangan *Noise*

Langkah ini bertujuan untuk menghilangkan seluruh *noise* yang dapat mempengaruhi output akhir yang akan dihasilkan oleh algoritma pengolahan data ini. Tahap ini dilakukan dengan menghilangkan objek yang memiliki luas kurang dari 175 pixel dengan menggunakan jenis konektivitas *four connected*. Jenis konektivitas ini menyatakan dua buah pixel saling berhubungan jika kedua pixel tersebut saling bersentuhan pada salah satu sisinya. Dengan menggunakan definisi tersebut maka suatu pixel akan memiliki hubungan dengan 4 pixel di sekitarnya yang saling bersentuhan pada sisinya. Pemilihan jenis konektivitas yang digunakan akan mempengaruhi jumlah obyek yang akan didapat dari perhitungan pada suatu *image*. Hasil langkah ini ditunjukkan padal Gambar 6b yakni berupa objek yang terletak dalam lingkupan cermin saja.

Untuk area di luar lingkup cermin, dilakukan langkah yang sama dengan batasan luas area lebih dari 3000 pixel dan hasilnya disajikan pada Gambar 6c. Luas objek dinyatakan dengan jumlah pixel yang ditempatinya. Pembandingan (subtraksi) kedua output ini menghasilkan suatu objek yang memang berada dalam lingkup tangkapan cermin cembung/oleh kamera, sehingga objek penelitian dapat diisolasi dari lingkungannya (Gambar 6d).

Penghitungan/Penentuan Objek

Tahap ini bertujuan untuk membuat fungsi yang menghasilkan *image* yang hanya memiliki sebuah objek saja. Tahap ini dilakukan dengan memisahkan beberapa objek yang mungkin ada pada satu *image* menjadi beberapa *image* yang hanya mengandung sebuah objek saja. Apabila hanya terdapat sebuah objek pada data *image*, maka *image* input akan sama dengan *image* output. Perintah khusus yang dirancang (dalam M-file) untuk menghitung jumlah objek ditunjukkan sebagai berikut:

[benda,jumlah] = hitungjumlahobjek(objek);

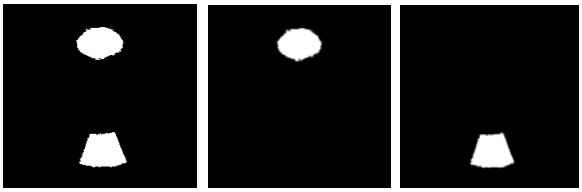
Cara kerja dari fungsi ini adalah dengan melakukan pemeriksaan setiap pixel yang ada pada *image* **objek** yang akan diberi label berbeda untuk setiap objek gambar yang berbeda.

Pelabelan dilakukan dengan memanfaatkan fungsi standar yang ada pada MATLAB. Fungsi ini akan memberi nilai 0 untuk latar belakang gambar, nilai 1 untuk objek pertama, nilai 2 untuk objek ke dua bila memang ada dan seterusnya. Contoh penggunaan fungsi tersebut adalah**:**

[semuaobjek,jumlah] = bwlabel(objek);

Perintah ini terdapat di dalam fungsi **hitung-jumlahobjek(),** dimana variabel **semuaobjek** yang disebut *cell arrays* akan menyimpan sejumlah *image* yang masing-masing berisi satu objek gambar saja. Sedangkan variable **jumlah** akan menyimpan nilai jumlah objek yang ada pada *image*.

Pada Gambar 7a, objek lingkaran akan memiliki label dengan nilai 1 dan objek kotak bernilai 2 sedangkan latar belakang hitam bernilai 0. Pertama-tama dilakukan pemeriksaan untuk objek bernilai label 1 (dalam **semuaobjek**). Apabila suatu pixel dalam variabel **semuaobjek**[5] bernilai sama dengan 1, maka nilai pixel itu akan disimpan dalam variabel **sementara**. Apabila tidak sama maka yang disimpan adalah nilai 0 yang menandakan lingkungan. Setelah seluruh pixel untuk objek pertama selesai diperiksa, maka variabel **sementara** akan dipindahkan ke variabel *array* **benda[]** pada *cell* 1 (yang menandakan objek pertama/objek lingkaran). Proses ini akan dilakukan lagi apabila terdapat lebih dari satu objek. Untuk objek kedua, nilai label adalah 2, pencarian dilakukan untuk nilai pixel sama dengan 2, hasilnya akan disimpan pada variabel *array* **benda[]** pada *cell* 2, dan begitu seterusnya. Gambar 7 menunjukkan contoh kasus dimana dalam suatu *image* awal terdapat dua buah objek, selanjutnya fungsi **hitungjumlahobjek()** ini akan membagi *image* tersebut menjadi 2 buah *image* yang masing-masing berisi satu objek saja, yakni lingkaran saja (Gambar 7b) dan kotak saja (Gambar 7c).



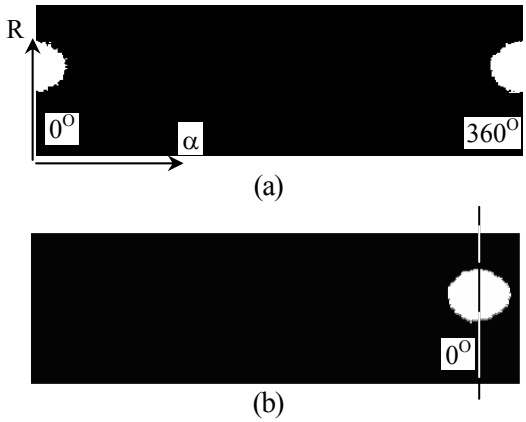
(a) (b) (c)

Gambar 7. *Image* Awal (a) dan *Image* Output dengan Objek Lingkaran (b) dan Kotak Saja (c)

Pembuatan *Panoramic Image*

Tahapan ini bertujuan untuk memperbaiki distorsi yang muncul akibat penggunaan cermin *spherical* (berbentuk bola). Output yang dihasilkan (*panoramic image/view*) nantinya dapat digunakan dalam menentukan arah objek terhadap sistem sensor, karena *panoramic view* memiliki salib sumbu arah sudut pandang terhadap jarak radial. Seperti ditunjukkan pada Gambar 3a, titik B memiliki koordinat (x,y) dan dapat dinyatakan dalam bentuk (α,R) dimana α dan R untuk setiap titik (x,y) dapat ditentukan.

Gambar 8a menunjukkan suatu benda lingkaran yang berada tepat di depan kamera (sumbu x) dan selanjutnya diolah menggunakan Pers. 1 dimana gambar tertangkap dalam layar akan berupa gambar setengah lingkaran putih masing-masing di sisi kiri dan kanan *panoramic view*. Bila image ini digeser beberapa derajat (secara sengaja), maka bentuk utuh objek tersebut dapat lebih jelas dilihat pada Gambar 8b, dimana garis vertikal menunjukkan garis batas kiri dan kanan yang telah disatukan pada *panoramic view* sebelumnya. Posisi garis vertikal ini menunjukkan titik awal scanning pada arah 0^O.



Gambar 8. *Panoramic Image/View* (a) dan Bentuk Utuh Objek Lingkaran (b)

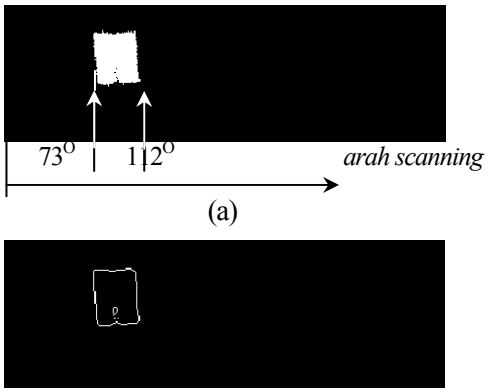
Penentuan Arah Objek dan Deteksi Tepian

Tahapan ini bertujuan untuk mendapatkan arah objek terhadap sistem sensor dan mendapatkan betuk tegas benda hasil tangkapan kamera. Arah benda tersebut merupakan salah satu komponen dalam menentukan keberadaan suatu objek yang ada disekeliling sensor vision ini. Arah objek dapat ditentukan dari *panoramic image* yang telah diperoleh pada langkah sebelumnya. Penentuan arah dilakukan dengan cara melakukan *scanning* secara horizontal bagi setip kolom pixel pada *panoramic image*. Nilai kolom batas antara objek dalam lingkungan merupakan arah objek yang dicari.

Langkah selanjutnya yang dilakukan adalah mengubah objek yang ada menjadi bentuk tepian garis. Langkah ini dilakukan karena input yang dibutuhkan dalam algoritma penentuan bentuk objek adalah garis tepi dari objek yang hendak dideteksi bentuknya. Pembuatan deteksi tepian dari objek dilakukan dengan menggunakan perintah *edge* yang terintegrasi dalam MATLAB dan metode yang digunakan adalah metode *canny*.

Sebagai contoh adalah suatu objek yang ditangkap oleh kamera berupa benda persegi panjang dan *panoramic image* yang dihasilkan ditunjukkan dalam Gambar 9a. Proses penentuan arah objek pada *panoramic image* memberikan hasil bahwa objek berada pada arah 73^0 sampai 112^0 karena sisi paling kiri benda mulai muncul pada nilai sudut $\alpha = 73^0$ dan

berakhir pada $\alpha = 112^0$. Perhitungan sudut dilakukan dengan rujukan sumbu x seperti disajikan pada Gambar 3. Proses pendeteksian tepian objek dari *image* tadi menunjukkan garis tepi dari objek yang tertangkap kamera seperti terlihat pda Gambar 9b. Bentuk benda dalam format garis tepian sering kali belum sempurna, sehingga langkah lanjut untuk penyempurnaan bentuk benda perlu dilakukan.



(a)

(b)

Gambar 9. Hasil *Scanning* Posisi Objek Persegi Panjang dalam *Panoramic Image* (a) dan Penyajian Objek dalam Bentuk Tepian (b)

Penentuan Bentuk Objek

Langkah selanjutnya yang dilakukan adalah menentukan bentuk dari objek yang hendak dikenali. Penentuan bentuk objek dilakukan dengan cara mencari jumlah titik sudut yang terdapat pada output dari algoritma deteksi tepian. Jumlah titik sudut tersebut akan dipakai untuk menentukan bentuk suatu objek. Algoritma penentuan bentuk objek dapat dibagi menjadi algoritma penentuan titik ujung garis lurus, penentuan titik sudut dan penentuan bentuk objek.

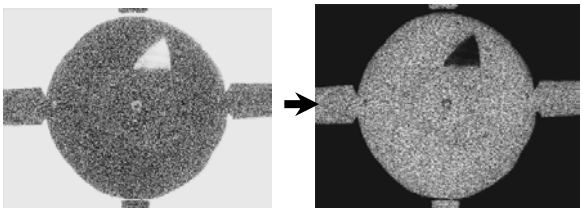
Algoritma ini dijalankan dengan cara melakukan *tracking* garis lurus secara vertikal pada *image* hasil deteksi tepian. Proses *tracking* dilakukan dengan pertama-tama mencari titik awal garis (pixel yang berwarna putih dan tiga pixel di atasnya yang berwarna hitam). Selanjutnya dilakukan *tracking* ke arah bawah sampai ditemukan titik akhir garis (pixel yang berwarna putih dan tiga pixel di bawahnya berwarna hitam).

Batas suatu garis dinyatakan sebagai garis lurus adalah kumpulan titik yang memiliki panjang lebih dari 15 pixel dan memiliki sudut lebih besar dari 149^0 (apabila dibuat vektor dari titik tengah garis ke titik awal garis dan dari titik tengah garis ke titik akhir garis). Batas ini diperlukan untuk dapat membedakan segmen garis lurus dan kurva pada lingkaran.

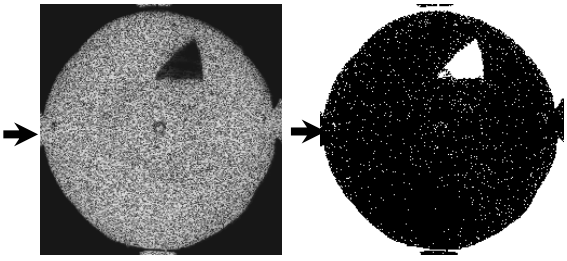
Selanjutnya dilakukan langkah penentuan jumlah titik sudut dari garis-garis lurus yang berhasil dideteksi. Titik sudut merupakan titik-titik awal dan akhir dari garis yang terdeteksi. Secara sederhana objek dibedakan berdasarkan jumlah titik sudut yang diperoleh. Objek disebut persegi panjang bila tedapat 4 titik sudut, segitiga bila terdapat 3 titik dan bila tidak terdapat titik sudut maka objek disebut lingkaran. Bila terdapat titik sudut diluar angka-angka tersebut maka dikatakan objek tidak dikenali oleh algoritma ini.

Sebagai contoh bekerjanya algortima pengenalan objek dengan sistem *omnidirectional image sensor* ditunjukkan pada Gambar 10. Setiap gambar yang ditunjukkan adalah hasil dari setiap langkah yang dikembangkan dalam penelitian ini. Langkah-langkah tersebut adalah:

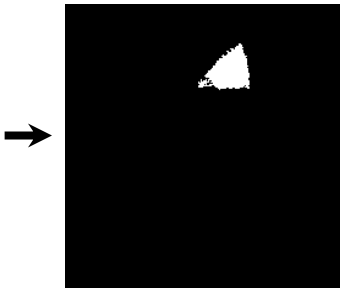
a. Gambar objek yang terlihat pada kamera (segitiga)
b. Pembacaan data *image* dari camera
c. *Image Cropping* untuk menghilangkan bagian tidak penting
d. Konversi *cropped image* menjadi *binary image*
e. Penghilangan noise dan isolasi objek
f. Pembuatan *panoramic image/view*
g. Pendeteksian tepi dan
h. Penentuan arah dan bentuk objek tangkapan kamera.



(a) (b)



(c) (d)



(e)

(f)

(g)

Gambar 10. Langkah Algoritma Secara Lengkap Penentuan Arah dan Bentuk Objek

KESIMPULAN

Sesuai dengan tujuan penelitian yang telah ditetapkan sebelumnya, maka dapat disimpulkan bahwa algoritma untuk memanfaatkan *omnidirectional image* dalam menentukan arah objek terhadap sistem sensor memerlukan beberapa tahapan, antara lain: pembacaan data *image*, kemudian *image cropping,* konversi *cropped image* menjadi *binary image,* penghilangan noise, penghitungan jumlah objek, pembuatan *panoramic image*, dan terakhir adalah penentuan arah objek terhadap sistem sensor.

Adapun langkah penentuan bentuk objek terdiri dari 2 algoritma yaitu: algoritma deteksi tepian dan penentuan bentuk objek. Setelah bentuk objek dapat diketahui, maka dari *panoramic image/view* dengan mudah dapat ditentukan arah di mana objek tersebut berada relatif terhadap salib sumbu sistem sensor ini.

Algoritma ini telah mampu dengan baik membedakan dan menentukan arah beberapa objek sederhana seperti persegi-panjang, lingkaran dan segi-tiga. Hasil penelitian ini masih perlu mendapatkan perbaikan terutama yang tekait dengan kesempurnaan *image* yang ditangkap kamera karena hal ini sangat dipengaruhi oleh kehalusan permukaan cermin, distrosi *image* yang cukup signifikan bila benda diletakkan pada dinding kotak dan lain sebagainya.

DAFTAR PUSTAKA

[1] Kim D. S., et.al. (1999). A Real Time Detection Algorithm for Direction Error in Omnidirectional Image Sensors for Mobile Robots. *IEEE International Conference on Robotics & Automation*, Detroit, Michigan.

[2] Young I. T., et. al. (1995). *Fundamentals of Image Processing*. Paperback, ISBN 90-75691-01-7

[3] Cao, et. al. (1986). Dynamic Omni-directional Vision for Mobile Robots. *Journal of Robotics Systems 3*, no. 1, p. 5-17.

[4] Nayar S. K., Peri V. (1999). Folded Catadioptric Cameras. *IEEE Computer Vision and Pattern Recognition Conference*, Fort Collins, USA.

[5] Bruckstein A., Richardson T. (2000). Omniview Cameras with Curved Surface Mirrors. *IEEE Workshop on Omnidirectional Vision* : Hilton Head, SC, USA.

[6] Nalwa V. (1996). *A True Omnidirectional Viewer*. Technical Report, Bell Laboratories, Holmdel, New Jersey, USA.

[7] McBride B., (2005) *Panoramic Cameras Time Line*. Online: http://www.panphoto.com/Time Line.html

[8] Gaspar J., et.al. (2005). *Toward Robot Perception Through Omnidirectinal Vision*. Online at: http://www.lkl.ac.uk/niall/book_odv.pdf

[9] Otsu N. (1979). A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics,* Vol. 9, No. 1, p. 62-66.