# *Object Detection System Sebagai Alat Bantu Mendeteksi Objek Sekitar untuk Penyandang Tunanetra.*

**Dr.Eng. Cahya Rahmad, ST. M.Kom.[1], Drs. Rawansyah, M.Pd.[2], Tanggon Kalbu Rochastu[3]**

[123]Program Studi Teknik Informatika, Jurusan Teknologi Informasi, Politeknik Negeri Malang
[1] cahya.rahmad@polinema.ac.id [2] rawansyah@polinema.ac.id [3]tanggonkalburochastu@gmail.com

***Abstrak*— Jumlah penyandang gangguan pengelihatan parah dan tunanetra di dunia ada sebanyak 216,6 juta orang dan 38,5 juta orang pada tahun 2018 dan akan meningkat setiap tahunya. Sedangkan perkembangan teknologi *Computer Vision* semakin populer karena adanya penggunaan sistem mengemudi otomatis menggunakan sistem deteksi objek. Teknologi tersebut bisa jadi solusi untuk membantu mereka. Hal ini dapat dilakukan dengan mengimplementasikan metode *Harris Corner Detection.* Metode *Harris Corner Detection* digunakan untuk mendeteksi sudut dari objek pada gambar yang diambil. Hasil deteksi tersebut akan dicari posisi dan jaraknya. Untuk jarak akan digunakan kaidah segitiga dalam mencari jaraknya. Sehingga nantinya dapat diprediksi lokasi dan jarak objek yang berada pada gambar yang diambil. Dari hasil implementasi diatas didapatkan bahwa akurasi deteksi objek menggunakan metode deteksi sudut *Harris Corner Detector* sebesar 88%. Sehingga aplikasi ini dapat melakukan deteksi objek berdasarkan sudut yang dideteksi dengan menggunakan *smartphone.***

***Kata kunci** : Object Detection, Corner Detection, Computer Vision, Tunanetra.*

## I. PENDAHULUAN

*Computer Vision* merupakan teknologi yang menyerupai kerja pengelihatan manusia[1]. Sehingga suatu mesin komputer bisa mengenali dan memahami suatu gambar. Salah satu aspek yang ada pada *Computer Vision* adalah *Object Detection*. Merupakan sistem yang bisa mendeteksi objek apa saja yang ada pada suatu gambar. Ada banyak metode yang bisa digunakan untuk mendeteksi objek. Metode yang paling cepat adalah metode YOLO : *You Only Look Once*. Metode ini lebih cepat dan lebih akurat dibandingkan metode seperti R-CNN dan Fast R-CNN yang notabene merupakan metode deteksi yang populer[2]. Metode tersebut merupakan deteksi objek bersamaan dengan identifikasi objek. Sedangkan teknologi yang hanya deteksi objek saja tanpa mengidentifikasi yaitu menggunakan *feature extraction*.

Teknologi deteksi objek ini bisa diterapkan di berbagai aspek untuk membantu navigasi karena cara kerjanya yang bisa menyerupai sistem pengelihatan manusia. *Feature extraction* merupakan tahapan awal dari deteksi objek[3]. Teknologi ini berkerja dengan cara mengambil fitur pada gambar. Gambar memiliki banyak fitur seperti contohnya garis, kontur, sudut dan warna. Fitur terebut penting sekali dalam bidang *image processing* dan *computer vision*[4]. Contoh metode ekstraksi fitur adalah *Harris Detector, Canny, Laplacian of Gaussian, Sobel* dan lain lain.

Pada beberapa tahun terakhir, sistem ini digunakan pada mobil sebagai alat bantu untuk navigasi ketika berkendara[5]. Menggunakan *edge detector* untuk mendeteksi garis markah jalan sebagai pedoman untuk navigasi berkendara dan *corner detector* untuk mendeteksi rambu lalu lintas[6]. Tidak hanya itu saja, sistem ini tentunya bisa bermanfaat khususnya bagi masyarakat yang memiliki gangguan pengelihatan karena fungsinya menyerupai pengelihatan manusia. Statistik mengatakan bahwa terdapat 216,6 juta orang yang memiliki gangguan pengelihatan dan sebanyak 38,5 juta orang yang menderita buta total yang diprediksi akan meningkat tiap tahunnya[7]. Hal ini menjadi tantangan yang harus dihadapi. Untuk menjaga keselamatan masyarakat tersebut, diperlukan akses terhadap informasi visual disekitarnya.

Terlepas dari cara yang sudah ada untuk membantu seperti asisten yang membantu orang yang memiliki gangguan pengelihatan, perlu dikembangkan sistem cerdas yang bisa secara otomatis membantu orang yang memiliki gangguan pengelihatan melakukan kegiatan sehari-harinya untuk membantu menavigasi dan mendeteksi objek - objek disekitarnya[8]. Maka dari itu diperlukannya untuk

membuat sistem yang mendeteksi objek disekitarnya pada platform android. Dengan menggunakan metode Harris Corner Detection yang akan diimplementasikan pada smartphone, pengguna dapat merasakan kemudahan untuk mendapatkan informasi disekitar mereka meskipun pengguna memiliki gangguan pengelihatan.

Dengan memanfaatkan kamera smartphone, pengguna dapat menekan tombol pada *headset/earphone* yang diprogram untuk mengambil gambar. Gambar yang diambil akan diproses menggunakan metode Harris Corner Detection sehingga mendapat informasi berupa teks mengenai apa saja yang tertangkap pada kamera. Informasi tersebut lalu dirubah menjadi bentuk audio sehingga pengguna bisa mendengarkan lokasi dan jarak objek didepanya. Sehingga diharapkan bisa membantu untuk mendeteksi objek disekitarnya dan membantu menavigasi para penggunanya.

## II. TINJAUAN PUSTAKA

### A. *Object Recognition*

*Computer vision* merupakan ilmu komputer dan sistem perangkat lunak yang dapat mengenali sebuah gambar. *Object recognition* merupakan bagian dari *Computer vision* yang paling populer. Hal ini dikarenakan o*bject recognition* merupakan aspek yang paling mendasar pada ilmu *computer vision.*

*Object Recognition* merupakan istilah kemampuan komputer untuk mengidentifikasi suatu gambar yang terdapat objek didalamnya[9]. Hal ini dilakukan dengan cara mengklasifikasikan suatu gambar ke dalam suatu kategori. Misalnya mengklasifikasikan antara gambar mobil dengan gambar sepeda motor ke dalam kategori mobil dan kategori sepeda motor.

### B. *Object Detection*

*Object Detection* atau deteksi objek merupakan bagian dari *Computer Vision. Object Detection* mengacu pada kemampuan komputer untuk mendeteksi sejumlah objek pada suatu gambar[9]. Hal ini dapat dilakukan dengan cara mengambil *image feature* seperti garis, sudut, kontur dan warna dari sebuah gambar[3]. Deteksi objek merupakan bagian dari *Object Recognition* atau identifikasi objek. Sehingga dapat disimpulkan bahwa untuk deteksi objek pasti harus diidentifikasi terlebih dahulu objek tersebut. Sedangkan pada penelitian ini hanya dilakukan deteksi objek saja tanpa adanya identifikasi objek.

### C. *Feature Extraction*

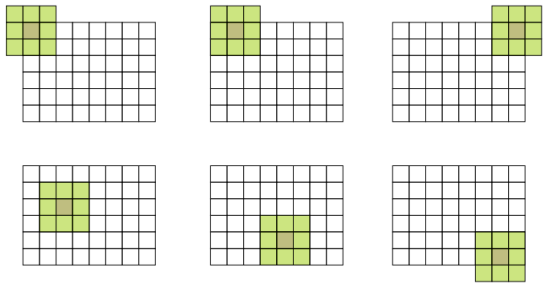
*Feature Extraction* merupakan ekstraksi fitur yang ada pada suatu gambar. Fitur yang dimaksud pada *computer vision* dan *image processing* adalah informasi penting pada gambar yang berguna untuk menyelesaikan masalah pada suatu tugas tertentu. Fitur merupakan hal yang sangat berkaitan dengan cara pengelihatan manusia berkerja. Fitur gambar yang paling mendasar adalah garis, sudut, kontur dan warna pada suatu gambar. Sudut merupakan garis yang saling berpotongan atau kontur yang berbeda dan saling bertemu. Hal ini merupakan langkah pertama dalam deteksi objek dan sangat penting dalam metode deteksi objek.

### D. *Convolution*

*Convolution* pada kamus berbahasa inggris Merriam-Webster memiliki definisi sebagai struktur perhitungan yang rumit. Tetapi pada *image processing, convolution* atau konvolusi ini memiliki makna sebagai proses menjumlahkan semua elemen gambar atau piksel pada suatu area tertentu yang dikali dengan *kernel.* Proses ini hampir sama seperti perkalian antar matriks. Perbedaanya prosesnya tidak menggunakan perkalian matriks pada umumnya, melainkan menggunakan perkalian titik atau biasa disebut dengan *dot product.* Berikut ini merupakan persamaan untuk perhitungan *dot product* dengan (x,y) adalah posisi filter dan (Tx,Ty ) adalah titik yang di filter*:*

$$H \otimes X = \sum_y \sum_x H(x,y)X(T_x - x, T_y - y) \qquad (1)$$

Dot Product berkerja dengan cara menjumlahkan semua nilai piksel yang ada pada ukuran tertentu. Ukuran tersebut bergantung pada ukuran *kernel* yang akan menjadi pembobotnya. Sehingga harus dilakukan iterasi agar keseluruhan gambar dapat dikonvolusi. Berikut ilustrasi perhitungannya:



Gambar 1. Ilustrasi Iterasi Perhitungan Konvolusi Kernel 3 x 3

| | | | | | |
|---|---|---|---|---|---|
| 100 | 100 | 10 | 0 | 0 | 0 |
| 100 | 100 | 10 | 0 | 0 | 0 |
| 100 | 100 | 10 | 0 | 0 | 0 |
| 10 | 10 | 10 | 0 | 0 | 0 |
| 10 | 10 | 10 | 0 | 0 | 0 |
| 10 | 10 | 10 | 0 | 0 | 0 |

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |

| | | | | | |
|---|---|---|---|---|---|
| 44 | 47 | 24 | 2 | 0 | 0 |
| 67 | 70 | 37 | 3 | 0 | 0 |
| 47 | 50 | 27 | 3 | 0 | 0 |
| 27 | 30 | 17 | 3 | 0 | 0 |
| 7 | 10 | 7 | 3 | 0 | 0 |
| 4 | 7 | 4 | 2 | 0 | 0 |

Gambar 2. Ilustrasi Gambar Sebelum dan Sesudah Konvolusi

Sehingga ketika iterasi dimulai akan ada nilai kosong pada semua pojok gambar sehingga pada gambar dibawah ini nilai tertingginya ada pada piksel di kolom kedua dan baris kedua. Hal tersebut dapat menyebabkan kekeliruan

informasi ketika penulisan kode program. Sehingga pada sisi gambar akan diberi nilai kosong untuk mencegah adanya kesalahan informasi. Ada banyak macam kernel yang bisa diugnakan untuk proses konvolusi antara lain:

TABEL I. CONTOH KERNEL

| No | Nama Kernel | Kernel |
|---|---|---|
| 1 | Blur | $\begin{bmatrix}1 & 1 & 1\\1 & 1 & 1\\1 & 1 & 1\end{bmatrix}$ |
| 2 | Sharpen | $\begin{bmatrix}1 & 2 & 1\\0 & 0 & 0\\-1 & -2 & -1\end{bmatrix}$ |
| 3 | Left Sobel | $\begin{bmatrix}-1 & 0 & 1\\-2 & 0 & 2\\-1 & 0 & 1\end{bmatrix}$ |
| 4 | Top Sobel | $\begin{bmatrix}1 & 2 & 1\\0 & 0 & 0\\-1 & -2 & -1\end{bmatrix}$ |

E. *Spatial Derivatives*

*Spatial Derivatives* adalah istilah yang digunakan untuk mencari nilai perubahan antara piksel satu dengan yang lain. Makna dari kata *derivative* sendiri adalah skala perubahan nilai pada suatu fungsi. Jika perbedaan diketahui maka bisa digunakan untuk mencari berbagai macam informasi pada pengolahan citra. Misalnya pada kernel Sobel yang digunakan untuk mencari garis pada gambar. Diketahui bahwa sobel menggunakan perbedaan piksel selanjutnya dikurangi dengan piksel sebelumnya.

$$M = \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix} = \begin{bmatrix} A & B \\ B & C \end{bmatrix} \quad (2)$$

F. *Eigen Value*

Merupakan nilai yang dilambangkan dengan simbol lambda ($\lambda$) yang digunakan sebagai patokan ketika mencari sudut. *Eigen Value* pada metode *Harris Corner Detector* berfungsi untuk mencari sudut yang berasal dari garis yang tidak horizontal ataupun vertikal.

$$\det M = \lambda_1 \lambda_2 = AC - BB \quad (3)$$

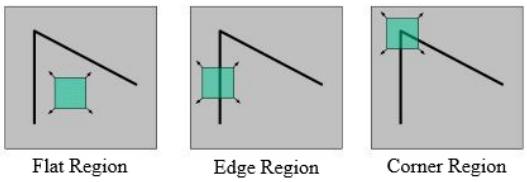
$$trace\ M = \lambda_1 + \lambda_2 = A + C \quad (4)$$

$$\lambda = \frac{traceM \pm \sqrt{traceM^2 - 4detM}}{2} \quad (5)$$

G. *Harris Corner Detection*

Harris Corner Detection merupakan metode yang bertujuan untuk mengambil dan mendeteksi sudut pada suatu gambar. Fitur ekstraksi sudut ini merupakan tahap awal dari deteksi objek ini. Deteksi sudut digunakan untuk mendeteksi sudut suatu objek pada gambar. Sehingga berfungsi untuk mendeteksi objek yang memiliki sudut seperti pintu, ujung bawah tembok, dan lain lain.

Metode ini berkerja dengan cara mempertimbangkan 8 piksel sebelahnya pada suatu pixel yang akan dideteksi sudutnya. Misalnya piksel p adalah piksel yang akan dideteksi sudutnya. Apabila 8 piksel di sebelahnya intensitasnya hampir sama dengan piksel p maka itu adalah *flat region.* Jika ada perubahan yang signifikan tetapi tidak ada perubahan pada piksel sekitarnya yang searah maka disebut dengean daerah garis atau *edge region*. Jika ada perubahan yang signifikan pada setiap arah maka piksel tersebut merupakan sudut. Berikut ilustrasi mengenai ide dari deteksi sudut :

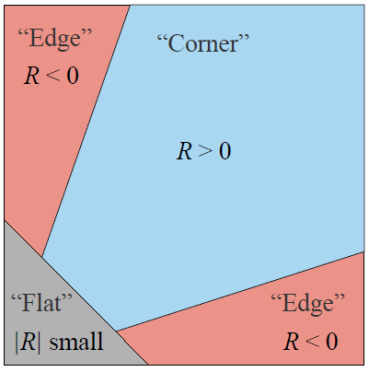



Gambar 3. Flat Region, Edge Region dan Corner Region

Langkah-langkah yang harus dilakukan pada metode ini adalah :

1. Memberi efek *grayscale* pada gambar asli. Hal ini bertujuan untuk mengurangi jumlah dimensi matriks dari 3 x 3 menjadi 2 x 2 agar mempermudah perhitungan.
2. Memberi efek *blur* yang bertujuan untuk mengurangi noise dari gambar yang diinputkan
3. Mencari *derivatives* dari matriks gambar yang diinputkan menggunakan perbedaan nilai piksel antara tetangganya seperti pada persamaan (2).
4. Nilai Derivatives disimbolkan dengan Ix dan Iy. Nilai tersebut akan digunakan pada persamaan (3) dan (4) untuk mencari determinan dan trace yang nantinya akan digunakan pada persamaan (5) untuk mencari nilai eigen.
5. Melakukan perhitungan Harris dengan rumus R = detM-k(traceM)$^2$ . Jika R memiliki nilai yang besar maka kemungkinan besar piksel tersebut adalah sudut

Jika R memiliki nilai yang rendah atau dibawah *treshold* yang ditentukan, maka area tersebut tergolong *flat region.*Jika R<0 yang berarti $\lambda_1 > \lambda_2$ atau sebaliknya maka area tersebut tergolong *edge region*. Jika R>0 dan skornya tingi maka $\lambda_1$ dan $\lambda_2$ bernilai tinggi maka area tersebut tergolong *corner region.* Ilustrasi untuk nilai R bisa dilihat pada gambar di bawah ini:



Gambar 4. Ilustrasi Nilai R

H. *Java*

Merupakan bahasa pemrograman yang berbasis *object oriented programming*. Bahasa pemrograman ini ditemukan oleh James Gosling pada tahun 1955. Bahasa ini banyak mengadopsi sintaks dari bahasa pemrograman C dan C++ sehingga sintaksnya mirip dan ada beberapa yang sama. Java merupakan bahasa yang implementasinya dibutuhkan dependensi seminimal mungkin. Hal ini menyebabkan java bisa berjalan di beberapa platform sistem operasi yang berbeda. Untuk melakukan pembuatan aplikasi pada java dibutuhkan JRE dan JDK. JRE merupakan singkatan dari Java Runtime Environment atau software yang digunakan untuk mengembangkan aplikasi berbasis java. Sedangkan JDK merupakan singkatan dari Java Development Kit. JRE merupakan bagian dari JDK. JDK merupakan kumpulan beberapa software yang terdiri dari JRE, compiler, dan *tools* lainya yang dibutuhkan untuk mengembangkan aplikasi berbasis Java

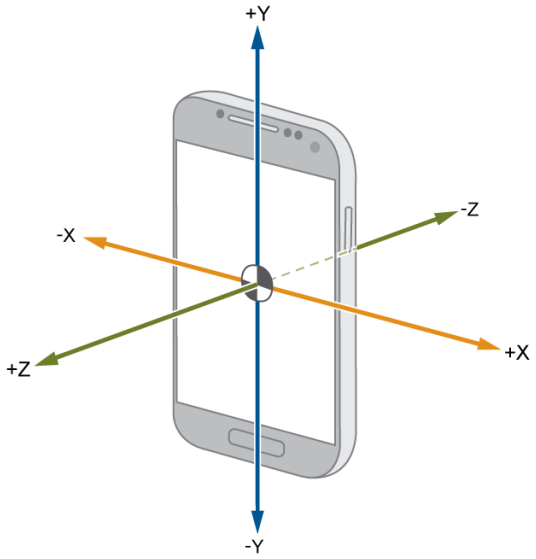
I. *Android Studio*

Android Studio merupakan IDE atau Integrated Development Environment milik Google yang digunakan untuk mengembangkan aplikasi berbasis Android. Android Studio menggunakan bahasa Java sebagai bahasa pemrogramannya. Android Studio memiliki beberapa fitur yang memudahkan para penggunanya seperti Gradle, Layout Editor dan lain lain.

J. *Accelerometer*

Accelerometer merupakan alat atau sensor yang bisa mengukur akselerasi yang tepat. Akselerasi yang dimaksud adalah perubahan kecepatan yang berpedoman pada gaya gravitasi bumi. Penggunaan accelorometer banyak digunakan pada segala bidang khususnya bidang yang membutuhkan sitem navigasi inersia. Seperti contohnya pesawat terbang yang menggunakan Accelerometer untuk medeteksi rotasi pesawatnya. Accelerometer juga digunakan pada *drone* untuk menjaga kestabilan ketika terbang. Pada *smartphone* juga dilengkapi dengan Accelerometer. Alat ini digunakan untuk mengetahui posisi rotasi *smartphone* yang. Contohnya digunakan untuk mengubah rotasi layar ke vertikal atau horizontal, mendeteksi motion atau gerakan ketika *smartphone* diangkat dan layar otomatis nyala, deteksi getaran, *games* dan lain lain.

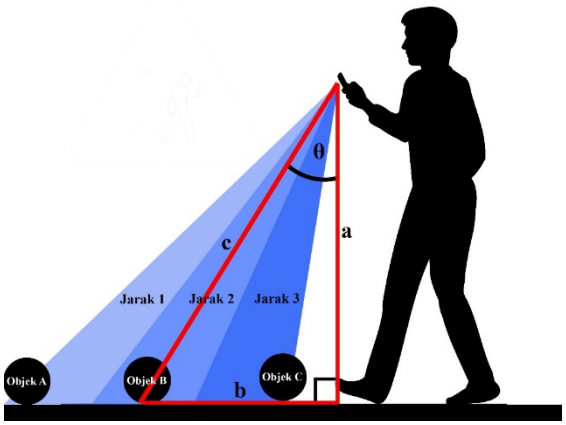
Pada penelitian ini Accelerometer digunakan untuk mendeteksi kemiringan *smartphone.* Kemiringan *smartphone* berfungsi untuk menghitung jarak objek dengan akurat menggunakan kaidah segitiga. Untuk menghitung kemiringan dapat dihitung dengan berdasar pada sumbu Y. Berikut ini merupakan ilustrasi arah kemiringan *smartphone* pada Accelerometer



Gambar 5. Sumbu pada Accelerometer
Sumber: mathworks.com

K. *Kaidah Segitiga*

Merupakan metode sederhana pada segitiga yang dapat digunakan sebagai pedoman untuk mecari jarak. Pada Gambar 2.6 merupakan ilustrasi seorang yang menggunakan smartphone. Variabel a merupakan jarak antara tanah dengan smartphone. Jarak tersebut dapat didapatkan dari 83% tinggi pengguna. Variabel b merupakan prediksi jarak antara pengguna dan objek yang dideteksi. Variabel c merupakan arah sudut pandang kamera. Sedangkan sudut Θ merupakan sudut yang dibentuk oleh sudut pandang kamera dengan jarak tanah dengan *smartphone*. Variabel Jarak 1, Jarak 2, dan Jarak 3 merupakan tiga golongan prediksi jarak. Objek A masuk pada golongan Jarak 1, Objek B masuk pada golongan Jarak 2, dan Objek C masuk pada golongan Jarak 3.




Gambar 6. Ilustrasi Segitiga

Jika pengguna memiringkan *smartphone* nya hingga sudut pandang kamera membentuk sudut $\theta$ maka bisa dihitung nilai b yang merupakan prediksi jarak antara pengguna dengan Objek B menggunakan persamaan (2). Jika Objek B dapat diketahui jaraknya, maka jarak Objek A lebih jauh dari Objek B. Jarak Objek C lebih dekat daripada jarak Objek B.

$$b = \mathrm{a} * \tan\theta \qquad (6)$$

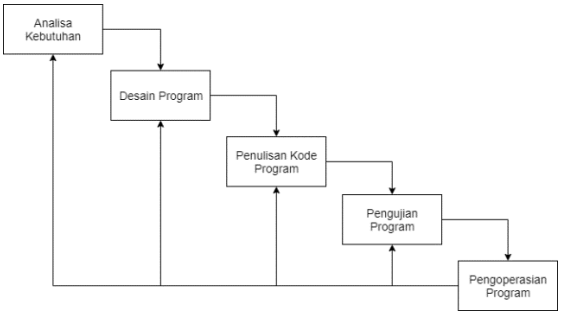
L. *Text to Speech*

*Text to Speech* atau kata ke suara merupakan fitur bawaan milik Google Android yang bisa langsung digunakan tanpa adanya tambahan *library* baru. Fitur ini berkerja dengan cara membaca tulisan yang ingin dibacakan menjadi suara yang bisa didengarkan. Sehingga bisa mengubah informasi berbentuk teks menjadi bentuk suara. Fitur ini dilengkapi berbagai bahasa yang bergantung pada jenis smartphone yang digunakannya. Bahasa Indonesia juga tersedia dalam fitur ini, tertapi kembali lagi tergantung pada jenis *smartphone* pengguna. Fitur ini menggunakan aktor suara yang sama seperti pada Google Translate, Google Talkback, dan Google Play Books. Fitur ini sangat penting pada penelitian ini karena media berbentuk suara merupakan alternatif untuk mengirimkan informasi ke pengguna yang menyandang tunanetra atau gangguan pengelihatan.

## III. METODOLOGI

A. *Metode Pengembangan Sistem*

Metodologi penelitian yang digunakan pada penelitian ini adalah model *Waterfall.* Model ini terdiri beberapa tahap yang beragam pada setiap pengembangan sistem tetapi harus memiliki fase analisa kebutuhan, desain program, pemrograman, pengujian dan penerapan program[10]. Penelitian ini menggunakan model yang terdiri dari tahap-tahap seperti pada gambar di bawah ini:




Gambar 7. Model Waterfall

B. *Metode Pengujian Sistem*

Untuk menguji keberhasilan ssitem dapat dilaukukan menggunakan beberapa cara, penelitian ini meggunakan tes akurasi dengan cara menghitung jumlah objek yang terdeteksi sama dengan objek yang ada. Selanjutnya menguji pengaruh cahaya terhadap akurasi sistem.

## IV. PERANCANGAN

A. *Deskripsi Sistem*

Sistem *Object Detection* untuk penyandang tunanetra ini merupakan sistem yang membantu menavigasi dan mendeteksi objek sekitar bagi mereka yang mengalami gangguan pengelihatan atau bahkan kebutaan. Sistem ini berkerja dengan cara mendeteksi sudut yang nantinya akan dianggap sebagai objek dan menginformasikan kepada pengguna melalui media suara.

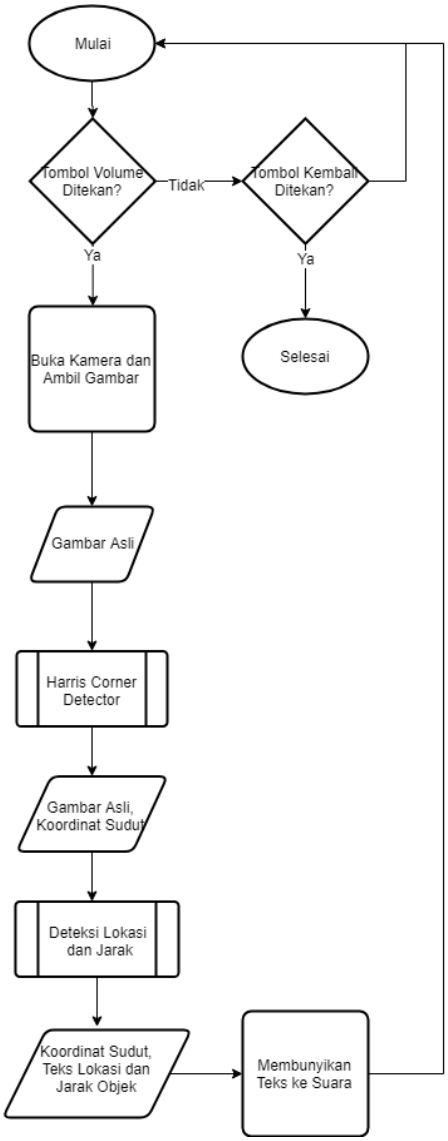
Kelebihan sistem ini dengan deteksi objek biasanya adalah sistem ini tidak memerlukan training objek yang ada. Sehingga segala jenis objek bisa terdeteksi. Tetapi deteksi tidak bersamaan dengan identifikasi atau klasifikasi gambar sehingga tidak bisa mengetahui benda apa yang terdeteksi.

B. *Kebutuhan Fungsional Sistem*

Kebutuhan fungsional adalah kebutuhan suatu sistem atau program yang berhubungan dengan fitur atau layanan yang diberikan oleh program. Berikut ini merupakan kebutuhan fungsional pada program ini:

1. Aplikasi dapat mengambil gambar dari kamera digital menggunakan tombol volume pada headset ataupun pada smartphone
2. Aplikasi dapat mendeteksi jumlah sudut yang ada pada gambar yang diambil oleh kamera digital
3. Aplikasi dapat mengetahui apakah ada objek dengan mengetahui jumlah sudut
4. Aplikasi dapat mengetahui posisi objek ada di kiri, kanan atau tengah gambar.
5. Aplikasi dapat mengidentifikasi objek apakah yang terdeteksi untuk beberapa objek tertentu
6. Aplikasi dapat menyuarakan mengenai informasi yang didapat mengenai gambar yang sudah diproses tersebut

C. *Perancangan Sistem*




Gambar 8. Flowchart Alur Program

Pada gambar diatas dijelaskan bahwa aplikasi tidak memiliki *user interface* ketika dimulai. Aplikasi sudah dimulai dan hanya menunggu *trigger* tombol volume. Ketika volume ditekan maka aplikasi akan membuka kamera dan mengambil gambar. Gambar tersebut nantinya akan diproses menggunakan *Harris Corner Detector* dan diambil kemungkinan ada atau tidaknya objek di gambar. Proses tersebut menghasilkan koordinat sudut dan teks mengenai ada atau tidaknya objek didepan. Setelah itu hasil koordinat tersebut akan diproses untuk mencari lokasi dan jarak objek tersebut menggunakan kaidah segitiga. Teks yang dihasilkan pada proses tersebut akan di parafrase dan dibunyikan sehingga pengguna bisa mendengar mengenai informasi lokasi dan jarak objek pada gambar yang diambil.

## V. IMPLEMENTASI

A. *Implementasi Program*

Program ini diterapkan menggunakan bahasa Java dan ditulis di Android Studi IDE. IDE merupakan kependekan dari *Integrated Developing Environtment*. IDE merupakan software yang menyediakan fitur untuk memngembangkan suatu software. IDE biasanya terdiri dari *text editor*, *compiler* dan *debugger*. Untuk implementasinya program akan di*install* di sistem operasi Android dengan minimal API 19 atau android KitKat.

B. *Pengambilan Gambar*

Pengambilan gambar merupakan proses awal pada aplikasi ini. Pengambilan gambar dilakukan menggunakan kamera yang tertanam pada *smartphone*. Sehingga diperlukan adanya izin untuk mengakses kamera. Untuk mengambil gambar, pengguna perlu menekan tombol volume naik ataupun turun yang berada di *smartphne* maupun di *earphone*.

C. *Preprocessing Gambar*

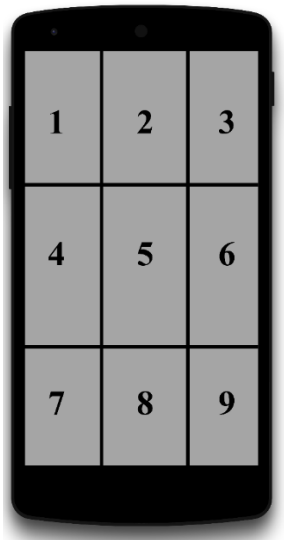
*Preprocessing* tersebut terdiri dari beberapa kegiatan yaitu *resize, grayscale dan Gaussian Blur. Resize* dilakukan untuk mengatur ulang ukuran gambar agar sesuai dengan ukuran layar dengan tujuan agar gambar tampil penuh ketika ditampilkan di layar. Hal ini berguna pada saat testing sebelum digunakan oleh pengguna aslinya yaitu para penyandang gangguan pengelihatan. Ketika sudah selesai pengujian maka tidak perlu di *resize* dan tidak perlu ditampilkan karena pengguna asli hanya membutuhkan informasi suara.

*Grayscale* diakukan dilakukan untuk meringankan beban perhitungan. Hal ini dikarenakan perhitungan *feature extraction* atau pengambilan *image feature* cukup dilakukan pada grayscale, perhitungan dengan warna akan menyebabkan redundansi[11]. *Gaussian Blur* dilakukan untuk mengurangi *noise* atau variasi warna yang tidak jelas pada gambar yang menimbulkan informasi yang tidak perlu.

D. *Proses Perhitungan Sudut*

1. Memberi efek *grayscale* pada gambar asli. Hal ini bertujuan untuk mengurangi jumlah dimensi matriks dari 3 x 3 menjadi 2 x 2 agar mempermudah perhitungan.
2. Memberi efek *blur* yang bertujuan untuk mengurangi noise dari gambar yang diinputkan
3. Mencari *derivatives* dari matriks gambar yang diinputkan menggunakan operator Sobel.
4. Melakukan perhitungan Harris dengan rumus $R = detM-k(traceM)^2$ . Jika R memiliki nilai yang besar maka kemungkinan besar piksel tersebut adalah sudut

E. *Penyajian Informasi*




Gambar 9. Ilustrasi Penentuan Informasi

Pada gambar penentuan jarak objek dilakukan dengan cara membagi gambar menjadi sembilan bagian sama rata. Pembagian dilakukan berdasarkan panjang dan lebar gambar. Sehingga area piksel yang dianggap sudut yang ditemukan pada area nomor satu memiliki jarak lebih dari jarak antara tanah dengan telepon genggam dan berlokasi disebelah kiri.

Untuk layar nomor lima memiliki jarak sama dengan jarak antara tanah dengan telepon genggam dan berlokasi didepan pengguna. Untuk layar kesembilan memiliki jarak kurang dari jarak antara tanah dengan telepon genggam dan berlokasi di kanan. Jarak ditentukan menggunakan kaidah segitiga sama kaki seperti pada persamaan nomor (1). Sudut yang diambil memiliki *range* antara 40-60 derajat. Penentuan sudut dilakukan menggunakan *Accelerometer.*

## VI. PENGUJIAN

A. *Pengujian Fungsional*

Pengujian Fungsional digunakan untuk melihat sistem sudah berjalan atau belum. Skenario pengujian dilakukan dengan menguji fitur-fitur dari aplikasi yang dirancang, mulai dari pengambilan gambar menggunakan tombol volume, melakukan perhitungan sudut, mengambil kesimpulan mengenai lokasi dan jarak objek, hingga menyuarakan informasi kepada penggunanya.

B. *Pengujian Akurasi*

Pengujian akurasi digunakan untuk melihat kemampuan sistem dalam mendeteksi sudut dalam suatu objek. Dalam pengujian ini dilakukan pengambilan gambar 15 kali secara acak di tempat yang acak juga. Hal ini bertujuan untuk mendeteksi objek yang bisa menjadi rintangan untuk para penyandang tunanetra dan gangguan pengelihatan. Untuk pengujianya dilakukan menggunakan sistemnya langsung sehingga tidak perlu aplikasi tambahan.

Pengujian dilakukan dengan cara mengambil gambar 50 kali pada tiap area. Setelah itu dicek kebenaranya menggunakan pengelihatan manusia dengan pengelihatan normal untuk mengetahi apakah variabel lokasi dan jarak objek tersebut pada aplikasi benar adanya. Akurasi dapat dihitung dengan menumlahkan jumlah prediksi benar lalu dibagi dengan jumlah keseluruhan prediksi.

Setelah dilakukan pengujian, didapatkan akurasi sebesar 88%. Angka tersebut diambil dari 44 kali benar dan 6 kali salah. Prediksi akan salah apabila banyak sekali noise dan adanya *pattern* yang tidak perlu pada lantai Hal ini mengakibatkan salah informasi. Aplikasi mendeteksi *pattern* lantai sebagai objek. Lantai atau lingkunan yang memiliki banyak lubang atau tidak rata atau memiliki *pattern* tertentu dapat menyebabkan terdeteksinya objek. Padahal tidak ada objek didepan yang menghalangi.

## VII. KESIMPULAN DAN SARAN

Berdasarkan hasil analisis, perancangan, implementasi, dan pengujian yang dilakukan terdapat beberapa kesimpulan yang didapat yaitu:

1. Mendeteksi objek menggunakan metode HCD pada *smartphone* dapat dilakukan menggunakan perhitungan manual dengan waktu proses dari pengambilan gambar sampai selesai berbicara selama 1 hingga 2 detik.
2. Memprediksi jarak objek dan memberikan informasi mengenai posisi objek dapat dilakukan menggunakan kaidah segitiga. Dengan memanfaatkan nilai tinggi pengguna dan tan sudut 45$^{o}$ sehingga didapatkan 3 kategori jarak yaitu 1,5 meter, lebih dari 1,5 meter dan kurang dari 1,5 meter.
3. Berdasarkan pengujian dengan benda nyata didapatkan akurasi sebesar 88% dengan jumlah prediksi benar 44 dan salah 6.

Berdasarkan penelitian yang dilakukan dapat diajukan beberapa saran sebagai berikut:

4. Untuk pengujian dalam luar ruangan dapat dikembangakan dengan menggunakan sensor jarak yang bisa menambah akurasi ada atau tidaknya objek pada tempat luar ruangan
5. Penggunaan sistem ini akan lebih baik jika dikembangkan dalam bentuk real time tidak memerlukan pengambilan gambar menggunakan tombol volume.

## DAFTAR PUSTAKA

[1] S. Milan and H. Vaclav, *Image Processing,*

*Analysis, and Machine Vision*. Thomson, 2008.

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, vol. 2016-Decem, pp. 779–788.

[3] K. Jeong and H. Moon, "Object detection using FAST corner detector based on smartphone platforms," in *Proceedings - 1st ACIS/JNU International Conference on Computers, Networks, Systems, and Industrial Engineering, CNSI 2011*, 2011.

[4] Y. Li, S. Wang, Q. Tian, and X. Ding, "A survey of recent advances in visual feature detection," *Neurocomputing*, 2015.

[5] R. Gandhi, "R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms," *Towards Data Science*, 2018. [Online]. Available: https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e. [Accessed: 04-Dec-2018].

[6] T. Wenzel, T. W. Chou, S. Brueggert, and J. Denzler, "From corners to rectangles-Directional road sign detection using learned corner representations," in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2017.

[7] R. R. A. Bourne *et al.*, "Magnitude, temporal trends, and projections of the global prevalence of blindness and distance and near vision impairment: a systematic review and meta-analysis.," *Lancet. Glob. Heal.*, vol. 5, no. 9, pp. e888–e897, 2017.

[8] F. S. Bashiri, E. LaRose, J. C. Badger, R. M. D'Souza, Z. Yu, and P. Peissig, "Object detection to assist visually impaired people: A deep neural network adventure," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018.

[9] K. Ikeuchi, "Computer Vision: A Reference Guide: I," in *Computer Vision: A Reference Guide*, 2014.

[10] X. Zhang, T. Hu, H. Dai, and X. Li, "Software development methodologies, trends, and implications," *Adopt. Is 2009 Model Curric. Apanel Sess. To Address Challenges Progr. Implement.*, 2009.

[11] M. A. Ruzon and C. Tomasi, "Edge, junction, and corner detection using color distributions," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2001.