ESTIMASI JARAK OBJEK PADA ARENA KONTES ROBOT PEMADAM API INDONESIA MENGGUNAKAN KAMERA DENGAN METODE *STEREOVISION*

SKRIPSI

Skripsi ditulis untuk memenuhi salah satu persyaratan dalam mendapatkan gelar sarjana teknik

Disusun Oleh :

ISKANDAR HUSEIN
3332111361

JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK UNIVERSITAS SULTAN AGENG TIRTAYASA
CILEGON – BANTEN
2018

PERNYATAAN KEASLIAN SKRIPSI

Saya menyatakan dengan sesungguhnya bahwa skripsi dengan judul:

ESTIMASI JARAK OBJEK PADA ARENA KONTES ROBOT PEMADAM API INDONESIA MENGGUNAKAN KAMERA DENGAN METODE *STEREOVISION*

Adalah hasil karya saya sendiri dan sejauh yang saya ketahui bukan merupakan tiruan atau duplikasi dari karya orang lain yang sudah dipublikasikan dan atau pernah dipakai untuk mendapatkan gelar kesarjanaan di lingkungan Universitas Sultan Ageng Tirtayasa atau di Perguruan Tinggi atau Instansi manapun, kecuali bagian yang bersumber informasinya dicantumkan sebagaimana mestinya.

Cilegon, Januari 2018

Iskandar Husein

NIM. 3332111361

PENGESAHAN PEMBIMBING

Skripsi dengan judul :

ESTIMASI JARAK OBJEK PADA ARENA KONTES ROBOT PEMADAM API INDONESIA MENGGUNAKAN KAMERA DENGAN METODE *STEREOVISION*

Dipersiapkan dan disusun oleh :

Iskandar Husein

3332111361

Dibuat untuk melengkapi sebagian persyaratan menjadi Sarjana Teknik pada Jurusan Teknik Elektro Fakultas Teknik Universitas Sultan Ageng Tirtayasa telah diperiksa dan disetujui oleh pembimbing skripsi.

Cilegon, 18 Januari 2018

Pembimbing 1	Pembimbing 2
Anggoro Suryo Pramudyo, M.Kom.	**Rian Fahrizal, S.T,M.Eng**
NIP. 198403042009121010	NIP. 197510262005011001

Mengetahui,

Ketua Jurusan Teknik Elektro

Dr. Supriyanto, S.T., M.Sc.

NIP. 197605082003121002

PENGESAHAN PENGUJI

Skripsi dengan Judul :

ESTIMASI JARAK OBJEK PADA ARENA KONTES ROBOT PEMADAM API INDONESIA MENGGUNAKAN KAMERA DENGAN METODE *STEREOVISION*

Dipersiapkan dan disusun oleh :

Iskandar Husein
3332111361
Telah dipertahankan di depan Dewan Penguji
Pada tanggal 18 Januari 2018

Susunan Dewan Penguji

Ketua	Anggota
Heri Haryanto S.T , M.T .	**H. Alief Maulana, S.T., M.T.**
NIP : 197611292003121003	NIP : 197401242009121001

Skripsi ini telah diterima sebagai salah satu persyaratan
untuk memperoleh gelar sarjana

Mengetahui,
Ketua Jurusan Teknik Elektro

Dr. Supriyanto, S.T., M.Sc.
NIP. 197605082003121002

ABSTRAK

Salah satu tugas penting dalam beberapa aplikasi pembuatan robot cerdas adalah kemampuan memperkirakan jarak benda, proses pengenalan objek pada robot sangat diperlukan pada beberapa aplikasi pembuatan robot cerdas. Dalam mendeteksi keberadaan objek di arena, robot perlu memiliki kemampuan untuk memperkirakan jarak benda. Berbagai cara dapat digunakan untuk mengetahui jarak benda, cara yang aktif yaitu dengan mengirimkan sinyal ke objek (sinar laser, sinar radio, *ultra sound*, dan lain sebagainya) sedangkan cara pasif yang dapat digunakan adalah dengan menggunakan citra stereo atau *stereovision* pada kamera Sistem yang dirancang menggunakan dua kamera dengan prinsip *stereovision*. Hasil Pengujian di arena Kontes Robot Pemadam Api Indonesia berhasil dengan cukup baik dengan keberhasilan deteksi objek di seluruh pengujian dan persentase kesalahan pengukuran sebesar 4,77 % dengan kesalahan tertinggi di Ruangan 3 dengan persentase kesalahan sebesar 7,45% dan tingkat kesalahan terendah di ruangan 4 sebesar 0,625%.

Kata kunci : *Stereovision,* Pengenalan Objek

ABSTRACT

One of the most important tasks in some intelligent robot-making applications is the ability to estimate the distance of objects, the process of recognizing objects in robots is necessary in some intelligent robot-making applications. In detecting the presence of objects in the arena, the robot needs to have the ability to estimate the distance of objects. Various ways can be used to know the distance of the object, the active way is by sending a signal to the object (laser light, radio light, ultra sound, etc.) while the passive way that can be used is to use stereo image or stereovision on camera. Testing results in the Indonesian Fire Brigade Robot Contest arena successfully performed well with the success of object detection throughout the test and the measurement error percentage of 4.77% with the highest error in Room 3 with the error percentage of 7.45% and the lowest error rate in room 4 of 0.625%

Keywords: *Stereovision, Object Recognition*

KATA PENGANTAR

بِسْـــــــــــــــمِ اللهِ الرَّحْمَنِ الرَّحِيْمِ

Assalamu'alaikum Wr. Wb

Alhamdulillah, segala puji dan syukur kehadirat Allah SWT, karena atas Rahmat dan Karunia-Nya, akhirnya penulis dapat menyelesaikan laporan tugas akhir ini dengan Judul "**Estimasi Jarak Objek pada Arena Kontes Robot Pemadam Api Indonesia Menggunakan Kamera dengan Metode *Stereovision***", disusun sebagai salah satu syarat dalam memperoleh gelar Sarjana Teknik Elektro di Universitas Sultan Ageng Tirtayasa.

Tugas Akhir ini diharapkan mampu memberikan manfaat dalam pengembangan Robot KRPAI agar mampu memperkirakan jarak terhadap objek sehingga mampu menyelesaikan pekerjaan lebih cepat sehingga Robot lebih kompetitif untuk mengikuti Kontes Robot Pemadam Api Indonesia, serta dapat dijadikan rekomendasi atau acuan pada penelitian yang lebih lanjut pada bidang terkait, dan juga dapat memperdalam pengetahuan mahasiswa tentang *Stereovision.*

Banyak pihak yang telah membantu dalam penyusunan laporan tugas akhir ini dalam berbagai hal seperti sarana dan pra sarana pendukung, data-data penting yang dibutuhkan, pemahaman materi berkaitan dengan tema yang penulis bahas, pada kesempatan ini penulis ingin menyampaikan rasa terima kasih yang sangat besar kepada :

1. Ayahanda tercinta almarhum Bapak H.Sugiarto, serta Ibunda tercinta Hj.Siti Asiyah, dan kedua adik tercinta yang telah memberikan segala doa dan dukungan baik materi maupun motivasi kepada penulis.
2. Bapak Dr. Supriyanto, S.T., M.Sc. selaku Ketua Jurusan Teknik Elektro Universitas Sultan Ageng Tirtayasa.
3. Bapak Ir.Ri Munarto, M.Eng selaku Dosen Wali Penulis. Yang telah memberikan banyak ilmu, saran serta bimbingannya selama ini.

4. Bapak Anggoro Suryo Pramudyo ,M.Kom. Selaku Dosen Pembimbing 1 yang telah memberikan banyak ilmu, saran, serta bimbingannya selama ini.
5. Bapak Rian Fahrizal ST, M.Eng. Selaku Dosen Pembimbing 2 yang telah memberikan banyak ilmu,saran, serta bimbingannya selama ini
6. Seluruh dosen Jurusan Teknik Elektro Universitas Sultan Ageng Tirtayasa, atas ilmu-ilmu yang telah diberikan kepada penulis selama ini, semoga Allah membalas semua kebaikan bapak dan ibu.
7. Teman-teman seperjuangan tugas akhir Abdul, Saturki, Bobby, Andreas, Arif Faishal, Faisal Ismi, Jaya Rachmat, Gilang Pratama serta seluruh teman-teman elektro 2011 semua yang selalu memberikan bantuan kepada penulis dalam penyelesaian laporan skripsi.
8. Teman-teman satu kontrakan penulis Mukhlisin, Maulana, Sobriansyah, Faisal Maulana, Keken, yang selalu memberikan bantuan dan motivasi kepada penulis yang sangat berharga dan bermanfaat.
9. Pihak-pihak yang telah banyak membantu baik secara langsung maupun tidak langsung yang tidak dapat disebutkan satu persatu.

Penulis menyadari sepenuhnya bahwa laporan skripsi ini masih jauh dari kesempurnaan, oleh karena itu dengan segala kerendahan hati penulis menerima kritik dan saran yang membangun demi kesempurnaan laporan ini. Akhirnya penulis berharap semoga skripsi ini dapat bermanfaat bagi pembaca semua.

Cilegon, Januari 2018

Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

BAB I
PENDAHULUAN

1.1. Latar Belakang

Dalam melaksanakan tugas memadamkan api, robot harus dapat menyusuri arena untuk dapat mencari dan memadamkan sumber api yang berupa lilin. Agar dapat menyusuri arena tersebut maka *mobile robot* yang dirancang harus mampu mendeteksi keberadaan dinding, *dog obstacle*, *furniture*, dan lorong yang menjadi lintasan robot [1]

Salah satu tugas penting dalam beberapa aplikasi pembuatan robot cerdas adalah kemampuan memperkirakan jarak benda. Dalam mendeteksi keberadaan objek di arena, robot perlu memiliki kemampuan untuk memperkirakan jarak benda. Berbagai cara dapat digunakan untuk mengetahui jarak benda, cara yang aktif yaitu dengan mengirimkan sinyal ke objek (sinar laser, sinar radio, *ultra sound*, dan lain sebagainya) sedangkan cara pasif yang dapat digunakan adalah dengan menggunakan citra *stereo* atau *stereovision* pada kamera [2].

Stereo Vision merupakan lingkup dari visi komputer. Titik dan tujuan akhir dari visi komputer adalah memungkinkan komputer dengan kemampuan "melihat" dan berfikir, yang sama atau bahkan menggunguli penglihatan manusia. *Stereo Vision* adalah teknik untuk mendapatkan sebuah impresi kedalaman dari alat penglihatan ke benda atau objek yang dituju dari alat tersebut. Teknik ini memungkinkan penggunaannya untuk menentukan jarak dari alat penglihatan ke benda yang dituju [3].

Beberapa tahun terakhir ini, terlihat perkembangan penelitian yang pesat pada berbagai bidang ilmu komputer, termasuk *computer vision*. Hal ini terjadi karena adanya minat dari para peneliti dan juga permintaan dari dunia industri dan masyarakat akan kemampuan baru yang dapat diberikan oleh komputer. Salah satu kemampuan yang paling diinginkan adalah rekonstruksi otomatis dan analisis lingkungan 3D serta rekognisi objek pada ruang tersebut. 3D *computer vision* ini dapat digunakan untuk melakukan navigasi otomatis dari robot dan kendaraan,

mengontrol lengan robot yang digunakan dalam industri, atau membuat model 3D dari suatu objek berdasarkan informasi 2D [4]

Stereoscopic adalah teknik yang digunakan untuk merekam dan mewakili Gambar *Stereoscopic* (3D). Hal ini dapat menciptakan ilusi kedalaman menggunakan dua gambar yang diambil pada posisi yang sedikit berbeda. Gambar *Stereoscopic* dapat diambil dengan sepasang kamera sehingga mirip dengan cara kerja mata manusia [5].

Dengan kemampuan memperkirakan jarak dapat mengurangi penggunaan sensor elektronis. Sistem dapat mengenali api dengan cepat sekaligus mendapatkan informasi jarak dengan penggunaan kamera stereo sehingga dapat mengenali objek lilin api dan mengukur jaraknya dari kamera.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas, maka dapat dirumuskan masalah penelitian ini sebagai berikut:

1. Bagaimana cara Sensor kamera menperkirakan jarak api pada arena Kontes Robot Pemadam Indonesia ?
2. Bagaimana cara mengolah data citra yang diterima dari sensor kamera sehingga dapat mengenali objek dan memperkirakan jarak dari kamera ?

1.3 Tujuan Penelitian

Tujuan dari penelitian yang akan dilakukan adalah sebagai berikut.

1. Melakukan pengenalan objek dengan menggunakan dua *webcam* yang datanya diolah dengan aplikasi *add-on* Cascade Training Detector yang terdapat dalam aplikasi MATLAB
2. Merancang sistem yang terdiri dari dua *webcam*, dan aplikasi MATLAB agar dapat mendeteksi objek dan memperkirakan jarak pada arena KRPAI

1.4 Batasan Masalah

Agar dalam penelitian ini tidak terlalu meluas untuk dilakukan, maka peneliti melakukan pembatasan dalam penelitian ini, sebagai berikut.

1. Objek yang dideteksi adalah lilin api pada arena sesuai *rules* Kontes Robot Pemadam Api Indonesia
2. Tidak membahas secara detail komponen-komponen penyusun perangkat keras
3. Jangkauan pengambilan citra terbatas sesuai dimensi simulasi robot buatan dari kayu dengan jangkauan maksimum pendeteksian 60 cm.
4. Proses pengenalan objek menggunakan metode *Viola-Jones Classifier*.

1.5 Manfaat Penelitian

Penelitian ini bermanfaat untuk pengembangan Robot KRPAI agar mampu memperkirakan jarak terhadap objek dan dapat mengenali objek dengan baik menggunakan kamera.

Manfaat bagi mahasiswa, khususnya mahasiswa elektro atau mahasiswa umum. Penelitian ini dapat dijadikan rekomendasi atau acuan pada penelitian yang lebih lanjut pada bidang terkait, dan juga dapat memperdalam pengetahuan mahasiswa tentang *Stereovision.*

1.6 Sistematika Penulisan

Sistematika penulisan skripsi ini dibuat dengan maksud memberi gambaran secara garis besar dari setiap bab dalam laporan skripsi ini.

BAB I PENDAHULUAN

Berisi mengenai pendahuluan, latar belakang permasalahan, perumusan masalah, pembatasan masalah, tujuan penulisan, metodologi penelitian dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini berisi tentang dasar teori Kontes Robot Pemadam Api Indonesia, *Stereovision*, dan metode Cascade Training.

BAB III METODOLOGI PENELITIAN

Bab ini berisi tentang metodologi penelitian serta perancangan penelitian.

BAB IV HASIL DAN ANALISA

Bab ini membahas hasil pengujian di arena Kontes Robot Pemadam Api Indonesia (KRPAI) serta analisa hasil pengujian.

BAB V KESIMPULAN

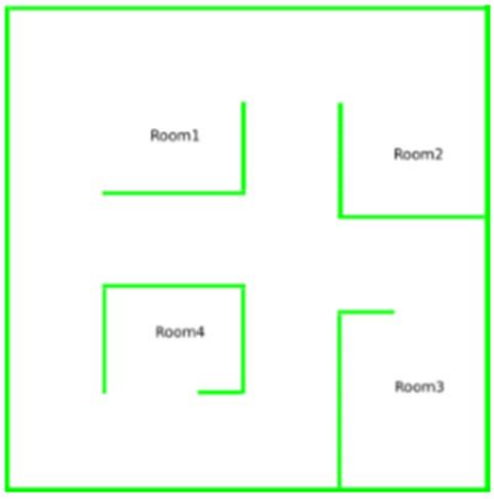
Membahas kesimpulan dari penelitian yang dilakukan serta saran untuk penelitian berikutnya.

BAB II
TINJAUAN PUSTAKA

2.1 Kontes Robot Pemadam Api Indonesia

Kontes Robot Pemadam Api Indonesia (KRPAI) merupakan salah satu kontes robot tingkat nasional yang diadakan oleh Direktorat Jenderal Pendidikan Tinggi secara teratur setiap tahun. Kontes robot KRPAI terbagi menjadi dua divisi yaitu divisi beroda dan divisi berkaki. Kedua divisi ini mempunyai tugas dan arena yang sama serta peraturan yang hampir sama. Dalam melaksanakan tugas memadamkan api, robot harus harus dapat menyusuri arena untuk dapat mencari dan memadamkan sumber api yang berupa lilin. Agar dapat menyusuri arena tersebut maka *mobile robot* yang dirancang harus mampu mendeteksi keberadaan dinding, *dog obstacle, furniture,* dan lorong yang menjadi lintasan robot [1].

Dalam mendeteksi keberadaan objek di arena, Robot perlu memiliki kemampuan untuk memperkirakan jarak benda. Berbagai cara dapat digunakan untuk mengetahui jarak benda, cara yang aktif yaitu dengan mengirimkan sinyal ke objek (sinar laser, sinar radio, *ultra sound,* dan lain sebagainya) sedangkan cara pasif yang dapat digunakan adalah dengan menggunakan citra *stereo* atau *stereovision* pada kamera [2].



Gambar 2.1 konfigurasi arena KRPAI 2017

Robot yang mengikuti Kontes Robot Pemadam Api Indonesia mempunyai tugas utama yaitu untuk memadamkan api yang terdapat pada arena pertandingan. Arena pertandingan pada Kontes Robot Pemadam Api Indonesia merupakan miniatur rumah. Robot diletakkan pada sebuah arena pertandingan, kemudian robot harus dapat menyusuri arena untuk dapat mencari dan memadamkan sumber api yang berupa lilin. Konfigurasi arena pertandingan yang digunakan dapat berubahubah sesuai dengan hasil undian. Robot harus mampu beradaptasi dan melaksanakan tugasnya sesuai dengan kondisi arena pertandingan.

2.2 Computer Vision

Computer vision adalah suatu ilmu di bidang komputer yang dapat membuat mesin atau *robot* untuk 'melihat' terdapat beberapa klasifikasi dari *vision* itu sendiri, yaitu *Low Level Vision, Medium Level Vision, dan High Level Vision*. *Low Level Vision* meliputi *Sensing*, yaitu pengambilan input berupa gambar, dan *Preprocessing*, yaitu memperoleh suatu gambar sebelum diproses. *Medium Level Vision* meliputi proses *Segmentation, Description, Recognition. Segmentation* adalah proses pemisahan gambar *digital* kedalam beberapa *region. Description* merupakan proses mendeskripsikan suatu gambar, sedangkan *Recognition* merupakan pengenalan terhadap suatu gambar. Pada level yang lebih tinggi (*High Level Vision*) terdapat proses *Interpretation, Interpretation* merupakan suatu kemampuan untuk memperkirakan bentuk asli dari gambar yang didapat, hal ini dapat dilakukan dengan cara mendapatkan berbagai informasi yang diperlukan pada gambar tersebut. Maka proses *Interpretation* memerlukan deteksi, indentifikasi, dan pengukuran dari fitur-fitur pada suatu gambar. [3] dalam hal ini *Stereovision* masuk dalam kategori *High Level Vision.*

Sebagai teknologi disiplin, *computer vision* berusaha untuk menerapkan teori dan sebagai model untuk pembangunan sistem *computer vision*. Contoh aplikasi *computer vision* mencakup sistem antara lain :

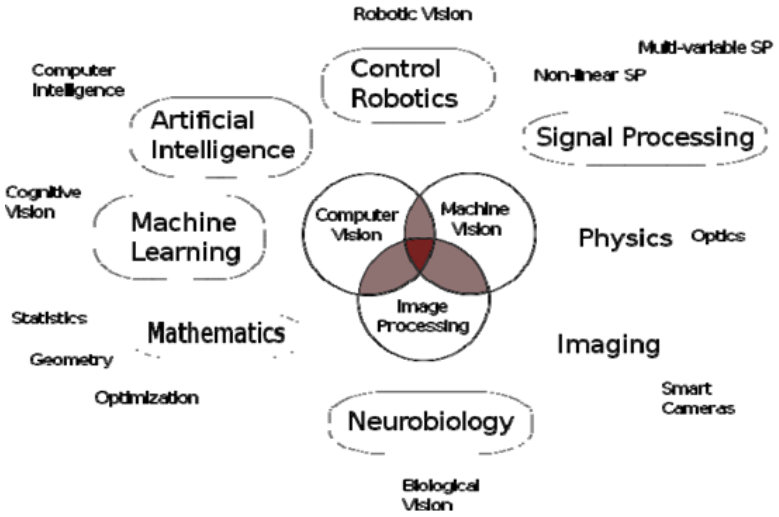
1. Pengendalian proses, misalnya : sebuah robot industri atau kendaraan otonom.

2. Mendeteksi kejadian (*Detecting events*), misalnya : untuk pengawasan visual dan *image sequences*.
3. Mengorganisir informasi, misalnya : untuk pengindeksan database foto dan citra urutan.
4. *Modeling objects or environments*, misalnya : industri inspeksi, analisis citra medis atau topografis model.
5. *Interaction*, misalnya : sebagai input ke perangkat untuk interaksi manusia komputer.

Computer vision berkaitan erat dengan studi tentang *biological vision* dan model proses fisiologis di balik persepsi visual pada manusia dan hewan lainnya. Di sisi lain, *computer vision* merupakan kajian dan menggambarkan proses yang dilaksanakan dalam perangkat lunak dan perangkat keras di belakang sistem penglihatan buatan. Interdisipliner pertukaran antara *biological vision* dan *computer vision* telah terbukti bermanfaat bagi kedua bidang.

Dalam beberapa hal *computer vision* merupakan kebalikan dari *computer graphics*. Sementara *computer graphics* menghasilkan data citra dari model 3D, *computer vision* sering menghasilkan model 3D dari data citra. Ada juga kecenderungan terhadap kombinasi dari dua displin ilmu tersebut (*computer vision* dan *computer graphics*). Sub-domain *computer vision* termasuk *scene reconstruction, event detection, video tracking, object recognition, learning, indexing, motion estimation, and image restoration*.

Hubungan antara *computer vision* dan berbagai bidang yang terkait diilustrasikan di gambar 2.2 berikut ini :



Gambar 2.2 Bidang-bidang *computer vision*

Banyak kecerdasan buatan (*artificial intelligence*) berkaitan dengan perencanaan otonom untuk sistem *robotical* untuk menavigasi melalui lingkungan. Informasi tentang lingkungan hidup dapat disediakan oleh sistem *computer vision*. Bertindak sebagai sensor dan visi tingkat tinggi yang menyediakan informasi tentang lingkungan dan robot. Kecerdasan buatan dan *computer vision* dalam bidang yang lain adalah *pattern recognition* dan teknik pembelajaran. Akibatnya, *computer vision* kadang-kadang dilihat sebagai bagian dari bidang kecerdasan buatan atau bidang ilmu komputer secara umum

Hal yang paling erat kaitannya dengan *computer vision* adalah pengolahan citra, analisis citra dan visi mesin. Ada tumpang tindih yang signifikan dalam berbagai teknik dan aplikasi ini. Ini berarti bahwa teknik-teknik dasar yang digunakan dan dikembangkan dalam bidang ini kurang lebih sama,.yang dapat diartikan sebagai sesuatu yang hanya memiliki satu bagian dengan nama yang berbeda. Di sisi lain, tampaknya sangat diperlukan bagi kelompok-kelompok penelitian, jurnal ilmiah, konferensi dan perusahaan untuk menampilkan atau sebagai pemilik dari salah satu bidang tersebut. *Computer vision* cenderung berfokus pada adegan 3D diproyeksikan ke satu atau beberapa citra, misalnya bagaimana merekonstruksi struktur atau informasi lain tentang adegan

3D dari satu atau beberapa citra. *Computer vision* sering bergantung pada satu atau lebih asumsi kompleks tentang adegan yang digambarkan dalam citra [4].

Sistem *Stereo Eyes* mempunyai konsep yang sama seperti 2 buah mata manusia yang terletak bersebelahan dengan jarak tertentu antara satu mata dengan yang lainnya. Ketika kedua mata tersebut melihat pada sebuah objek yang sama, masing-masing mata menangkap gambar dengan sudut pandang yang berbeda dan kemudian gambar tersebut dikirim ke otak untuk diproses. Ketika kedua gambar yang ditangkap oleh kedua mata tiba secara bersamaan di belakang otak, mereka bersatu menjadi satu gambar. Penglihatan tersebut dinamakan dengan penglihatan *stereo* (*stereo vision*). Pikiran manusialah yang menggabungkan kedua gambar dengan mencocokkan bagian yang sama dan menambahkan sedikit perbedaan sehingga dapat memberikan informasi dari suatu gambar, salah satu informasi yang dapat diberikan adalah posisi objek dalam gambar tersebut. Salah satu metode yang digunakan untuk mendapatkan informasi posisi ini adalah perhitungan *disparity* (perbandingan 2 gambar) dari gambar *stereo* (*stereo image*) yang didapatkan dari penglihatan *stereo* (*stereo vision*). Meskipun begitu, metode perhitungan *disparity* hanya akan memberitahukan posisi objek tersebut terhadap objek lainnya tanpa informasi jarak. Informasi jarak merupakan perkembangan selanjutnya dari *computer vision* menggunakan *stereo vision* [3]

2.3 Stereovision

Stereo vision menggunakan dua kamera untuk mengamati lingkungan, menemukan titik yang sama di setiap citra, dan mengukur kedalaman pada titik itu dengan triangulasi, yaitu dengan melihat garis - garis yang berpotongan dari setiap kamera terhadap objek. Menemukan titik yang sama disetiap citra disebut korespondensi yang merupakan tugas yang mendasar dari *computer stereo vision*.

Depth perception adalah proses ekstraksi kedalaman setiap pixel dalam citra yaitu, menemukan dimensi ketiga dalam citra, dilambangkan dengan (z). Sebuah citra 3D memiliki banyak keunggulan dibandingkan dengan citra 2D. Citra 2D hanya memberikan informasi yang terbatas tentang bentuk dan ukuran fisik suatu objek. Sementara citra 3D mengungkapkan geometri dalam bentuk koordinat 3D, oleh karena itu ukuran dan bentuk dari sebuah objek dapat dihitung

dari koordinat 3D. [4]

Metode untuk memperoleh informasi kedalaman menggunakan stero vision, salah satunya adalah Curve Fitting [3] dalam berbagai aktifitas dalam dunia sains, seringkali diperlukan suatu fungsi yang menghubungkan antar *variable-variable* dari data yang diamati. Untuk itu diperlukan suatu metode untuk menentukan bentuk kurva yang merupakan representasi dari data tersebut sehingga bisa dipergunakan untuk memprediksi pola atau kecenderungan dari data yang diamati. Selain itu kurva tersebut dapat dipergunakan untuk mencari nilai suatu titik di antara nilai-nilai yang diketahui (diamati) [4].

Metode ini menggunakan pendekatan look-up table untuk memperoleh informasi kedalaman. Dengan kata lain diperlukan sebuah database yang menyimpan sekumpulan informasi mengenai hubungan antara kedalaman dan jumlah *pixel* tertentu dari objek. Database diperoleh dengan melakukan proses sampling pada jarak-jarak tertentu dan menghitung jumlah pixel yang ada. Setelah database terbentuk, *curve fitting* dilakukan untuk memperoleh sebuah kurva atau persamaan matematika yang paling tepat untuk menggambarkan sekumpulan data yang ada. Hal ini mungkin untuk diterapkan karena jumlah *pixel* objek akan semakin berkurang seiring bertambahnya nilai kedalaman. Persamaan yang digunakan untuk melakukan curve fitting adalah fungsi exponential, *power law* dan *polynomial 4th*. Dari hasil percobaan diperoleh bahwa curve fitting dengan *polynomial 4th* menghasilkan error yang lebih kecil dibandingkan dengan persamaan *curve fitting* yang lain. Selain itu diperoleh bahwa keakurasian dari metode ini sangat bergantung pada kondisi pencahayaan pada saat pengambilan gambar [3]

Mengetahui perkembangan algoritma pada computer vision yang begitu pesat dan harga kamera yang semakin terjangkau, peneliti menerapkan stereo vision system yang menggunakan kamera sebagai sensor dan meniru cara kerja dari mata manusia untuk memperoleh informasi 3D. Informasi ini nantinya digunakan untuk mendeteksi dan mengestimasi posisi objek. Keunggulan penggunaan kamera sebagai sensor dibandingkan dengan laser atau gelombang radio adalah

diperolehnya informasi mengenai warna objek yang dapat digunakan sebagai salah satu faktor penentu dalam proses rekognisi objek [5].

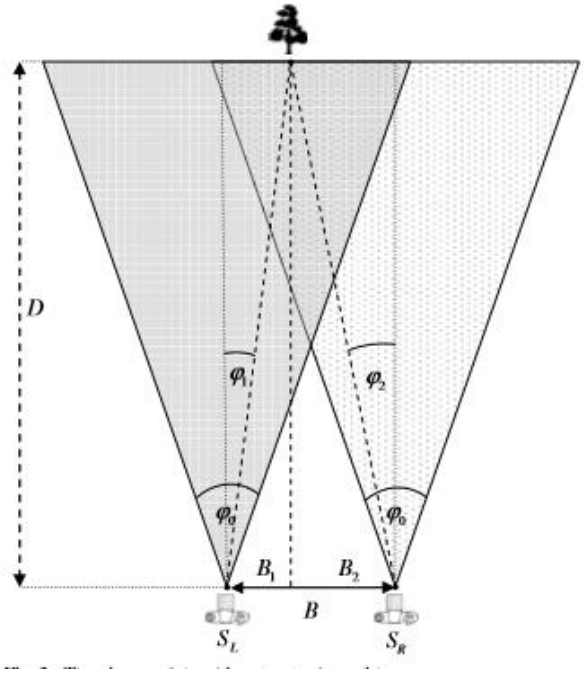
Stereoscopic adalah teknik yang digunakan untuk merekam dan mewakili Gambar Stereoscopic (3D). Hal ini dapat menciptakan ilusi kedalaman menggunakan dua gambar yang diambil pada posisi yang sedikit berbeda. Gambar Stereoscopic dapat diambil dengan sepasang kamera sehingga mirip dengan cara kerja mata manusia [6].

Metode Euclidean adalah metode pengukuran jarak garis lurus (straight line) antara titik X (X1, X2, …Xn) dan titik Y (Y1, Y2, ….Yn)., yaitu berupa garis lurus ,metode Euclidean sendiri memiliki rumus (formula) pengembangannya sesuai dengan keadaan ruang. Untuk menghitung metode Euclidian dapat menggunakan persamaan (2.1) dibawah ini :

$$\text{Dist} = \sqrt{\sum_{k=1}^{n}(p_a - q_a)} \quad (2.1)$$

Keterangan :
n = jumlah dimensi atau atribut
p_a = atribut ke –a dari objek p.
q_a = atribut ke-a dari objek q

Metode lainnya yaitu *Geometrical Derivations*, yang memperoleh informasi kedalaman objek menggunakan persamaan yang diturunkan berdasarkan pada geometri dari *stereo vision,* Hal ini dapat menciptakan ilusi kedalaman menggunakan dua gambar yang diambil pada posisi yang sedikit berbeda. Ada dua cara yang mungkin untuk mengambil gambar stereoskopik dengan menggunakan kamera stereo dua lensa khusus atau sistem dengan dua kamera lensa tunggal digabungkan [6].



Gambar 2.3 Geometri *Stereovision*

Dari gambar di atas didapatkan persamaan sebagai berikut :

$B = B_1 + B_2 = D \tan\phi_1 + D \tan\phi_2$

$$D = \frac{B}{\tan\phi_1 + \tan\phi_2} \tag{2.2}$$

$$\frac{x_1}{\frac{x_0}{2}} = \frac{\tan\phi_1}{\tan(\frac{\tan\phi_0}{2})}$$

$$\frac{-x_2}{\frac{x_0}{2}} = \frac{\tan\phi_2}{\tan(\frac{\tan\phi_0}{2})} \tag{2.3}$$

Dari Persamaan (2.2) dan (2.3) diperoleh

$$D = \frac{Bx_0}{2\tan\left(\frac{\varphi_0}{2}\right)(x_L - x_D)} \tag{2.4}$$

Keterangan :

D = Kedalaman Objek dari kamera

B = Jarak antar kedua kamera

Φ_0 = *horizontal angle of view*

$|(x_L - x_D)$ = *disparity*

x_0 = Resolusi gambar

2.4 Pengenalan Objek

Pengenalan pola (*pattern recognition*) adalah ilmu yang mempelajari cara untuk mengenali suatu objek atau pola dengan menggunakan komputer. Pola yang ingin dikenali harus memiliki ciri yang spesifik dan di dalam satu himpunan *class*, objek harus sejenis tetapi tidak harus identik sehingga dapat dibedakan dengan objek pada *class* lain.

Kemampuan untuk mengenali pola adalah dasar visi komputer. Istilah pola di sini adalah sebuah deskripsi struktural dan kuantitatif dari suatu objek. Secara umum, satu atau beberapa deskriptor membentuk pola. Bidang pola berkorespondensi dengan sebuah pengukuran atau sebuah bidang observasi. Vektor pola terkadang dirujuk sebagai vektor observasi. Vektor pola seringkali berisi informasi yang berlebih-lebihan. Oleh karena itu, vektor pola dipetakan ke sebuah vektor fitur. Sistem pengenalan pola biasanya mempertimbangkan akan dipetakan terlebih dahulu ke vektor fitur manakah sebuah bidang fitur. Vektor fitur digunakan untuk memutuskan termasuk kelas manakah suatu *sample* input. Tujuan dari pengekstraksian fitur adalah untuk mengurangi data dengan memperoleh fitur atau properti pasti yang membedakan pola input.

Salah satu pendekatan yang terkenal untuk merepresentasikan batasan-batasan adalah tanda tangan. Salah satu cara sederhana untuk menghasilkan tanda tangan adalah dengan memetakan jarak dari pusat ke batasan sebagai suatu fungsi dari suatu sudut. Pemetaan ini mengurangi representasi batasan menjadi sebuah fungsi satu dimensi yang dapat didigitisasikan menjadi sebuah vektor. Deskriptor seperti kode rantai (*chain code*) dan rasio area dengan keliling digunakan untuk mengkarakterisasi bentuk [8].

2.4.1 *Viola-Jones Classifier*

Pada dasarnya pengolahan gambar yang standar akan mengubah skala gambar untuk ukuran yang berbeda kemudian menjalankan detektor dalam bentuk gambar-gambar yang sudah dalam ukuran yang tetap. Karena pendekatan tersebut memakan waktu yang cukup lama dalam perhitungan gambar yang mempunyai ukuran yang berbeda. Maka pada tahun 2001, Paul Viola dan Michael Jones telah menyusun skala detektor *invariant* yang membutuhkan jumlah yang sama dalam melakukan perhitungan pada ukuran gambar yang berbeda. Algoritma ini disebut juga dengan *Viola-Jones*. Algoritma detektor *Viola-Jones* mempunyai 3 fitur [9] yaitu:

1. *The Scale Invariant detector*

   Pada tahap ini, hal pertama yang dilakukan adalah mengubah citra gambar ke dalam bentuk integral. Hal ini dilakukan dengan membuat setiap piksel mempunyai nilai yang sama dengan jumlah keseluruhan semua piksel yang berada di atas dan di kiri dari piksel yang bersangkutan.

1	1	1
1	1	1
1	1	1

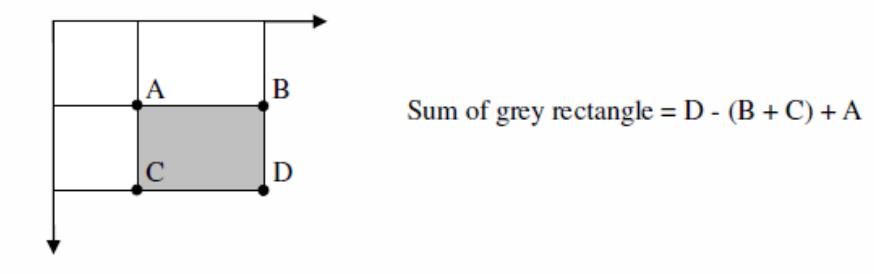
Input image

1	2	3
2	4	6
3	6	9

Integral image

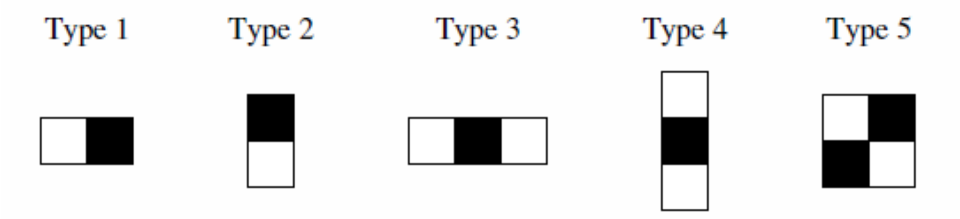
Gambar 2.4 Gambar Integral

Jika *integral image* sudah dilakukan maka untuk melakukan perhitungan jumlah seluruh piksel yang berada di dalam persegi panjang hanya menggunakan empat nilai yang bertepatan dengan sudut persegi panjang [8]



Gambar 2.5 Perhitungan pada gambar integral

Dalam melakukan analisis perhitungan jumlah seluruh piksel, *Viola-Jones* memberikan *sub-window* yang berukurang 24*24 piksel dengan menggunakan dua atau lebih persegi panjang.Setiap hasil yang didapatkan di hitung dari pengurangan antara jumlah piksel yang putih dengan jumlah piksel yang hitam. [9]



Gambar 2.6 Sub-window *persegi panjang*

2. *The modified AdaBoost algoritm*

   Metode yang dibangun oleh *Viola-Jones* merupakan modifikasi dari algoritma *AdaBoost* yang dibuat oleh Freund dan Schapire pada tahun 1996. *AdaBoost* adalah algoritma *machine learning boosting* yang mampu membuat sebuah *classifier* kuat melalui kombinasi berat pada *classifier* lemah. Dalam matematika *classifier* lemah di representasikan sebagai berikut

$$h(x, f, p, \theta) = \begin{cases} 1 \; jika \; pf > p\theta \\ 0 \; selainnya \end{cases} \quad (2.5)$$

   Keterangan : x adalah 24 x 24 piksel *sub-window*

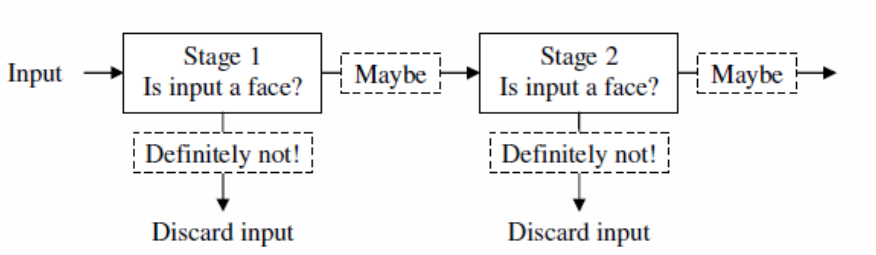
   f adalah fitur yang diterapkan

   p adalah fitur yang berlawanan

θ adalah*threshold* yang memilih nilai x dalam menentukan sebuah wajah atau bukan

3. *The cascade classifier*

   Prinsip dasar dari algoritma *Viola-Jones* adalah mencari detektor sebanyak mungkin dalam satu gambar yang sama. Metode ini digunakan untuk menentukan bagian yang bukan objek dan bagian objek. Ketika *sub-window* menyatakan bukan bagian wajah maka secara automatis bagian tersebut dihilangkan dan mencari ke bagian yang lainnya.Metode ini melakukannya dalam beberapa tahap dengan memberikan *sub-window* [9]



Gambar 2.7 Proses *Cascade Classifier*

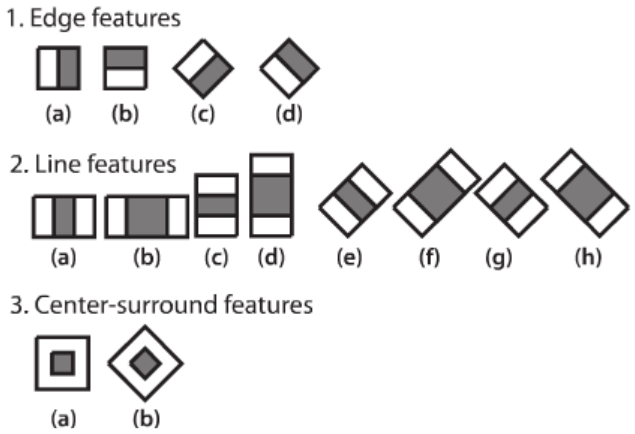
Kerangka *Viola-Jones* telah banyak digunakan oleh para peneliti untuk mendeteksi lokasi objek dalam citra yang diberikan. Pengklasifikasi *fitur input* deteksi objek dikenal oleh banyak peneliti, seperti *OpenCV*

Haar Cascade Classifier menggunakan AdaBoost di setiap node dalam kaskade untuk mempelajari tingkat deteksi tinggi dengan *multi-tree classifier* penolakan pada setiap node [8]

Algoritma ini menggabungkan beberapa fitur inovatif, seperti:

1. Menggunakan *haar-like, threshold* yang digunakan untuk jumlah dan membedakan daerah persegi dari gambar.
2. Teknik gambar Integral yang memungkinkan perhitungan cepat untuk daerah persegi atau wilayah yang diputar 45 derajat. Struktur data ini digunakan untuk membuat perhitungan dari fitur input Haar-like lebih cepat.

3. Banyaknya proses training Meningkatkan untuk membuat klasifikasi simpul biner (ya / tidak), ditandai dengan tingkat deteksi yang tinggi dan tingkat penolakan lemah.
4. Pengorganisasian node classifier kurang baik dari kaskade penolakan. Dengan kata lain, kelompok pertama dari pengklasifikasi dipilih sehingga deteksi terbaik di wilayah citra terdiri dari obyek meskipun memungkinkan banyak kesalahan dalam deteksi. Kelompok *classifier* berikutnya adalah deteksi terbaik kedua dengan tingkat kesalahan sedikit dan seterusnya. Dalam pengujian, sebuah objek dapat diketahui jika objek yang membuatnya melalui semua *cascades*. Fitur *input* Haar-seperti yang digunakan oleh classifier adalah:



Gambar 2.8 Fitur Input Haar

BAB III
METODOLOGI PENELITIAN

3.1 Metodologi Penelitian

Metode penelitian yang digunakan skripsi ini adalah sebagai berikut:

1. Studi Literatur

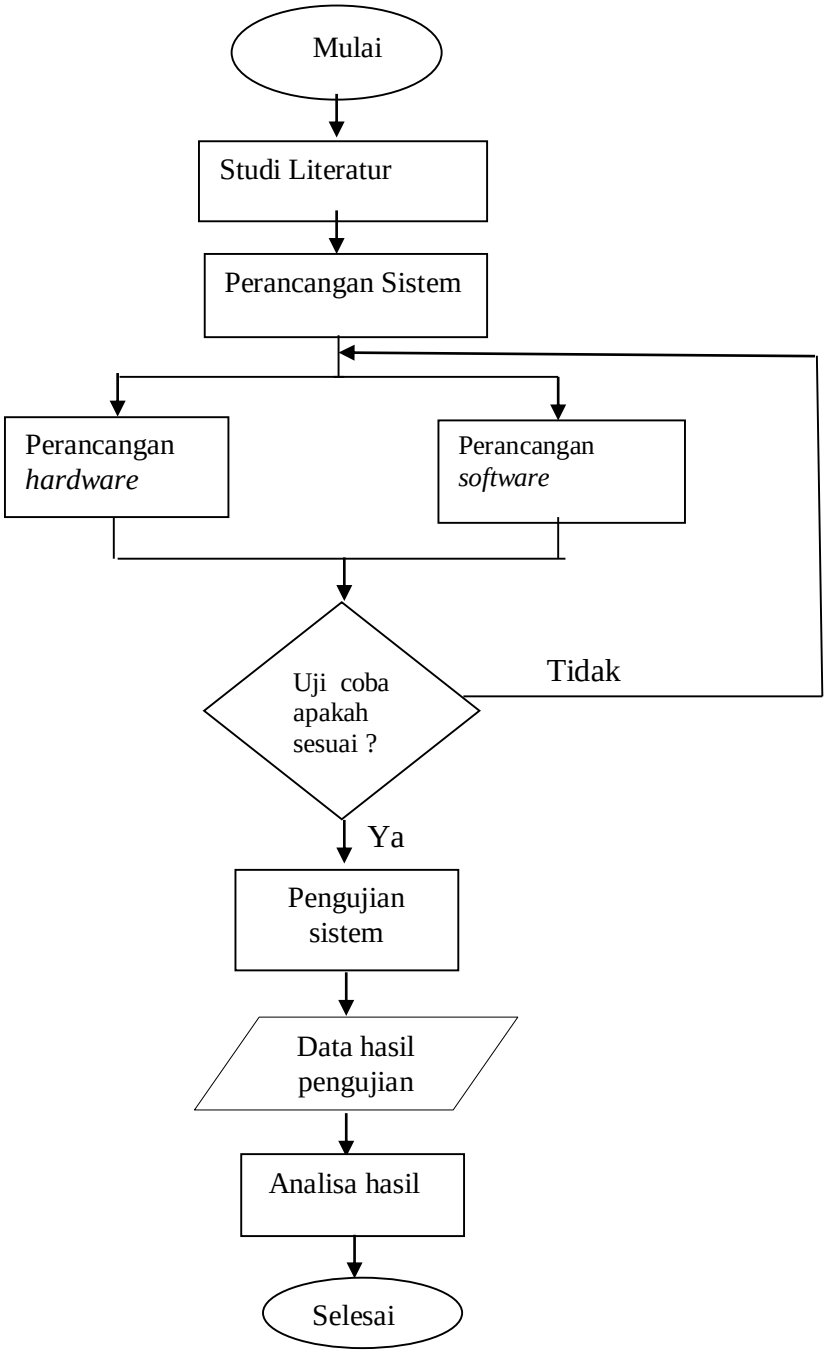
   melakukan studi literatur dengan referensi berupa jurnal penelitian baik jurnal nasional maupun jurnal internasional, buku-buku, dan artikel-artikel terkait dengan pokok permasalahan yang dibahas dalam skripsi ini.
2. Perancangan Penelitian

   Untuk melakukan penelitian ini terlebih dahulu dilakukan Perancangan *hardware* berupa dua buah *webcam*, simulasi Robot dan perangkat Laptop Serta perancangan *software* berupa program stereovision di Matlab R2017a
3. Pengujian sistem

   Pengujian sistem dilakukan di arena Kontes Robot Pemadam Api Indonesia (KRPAI) , dengan meletakkan objek lilin di ruangan yang ada dan mengambil gambar untuk diolah di Matlab R2017a kemudian mendeteksi objek serta menghitung jarak dari kamera
4. Analisa hasil

   Setelah selesai pengujian data pengukuran jarak dengan metode Stereovision kemudian dilakukan analisa

Untuk mencapai tujuan penelitian dilakukan dalam tahapan – tahapan di atas , berikut ini gambaran *flowchart* penelitian yang dilaksanakan



Gambar 3.1 *Flowchart* penelitian

3.2 Instrumen Penelitian

Perangkat yang digunakan pada skripsi ini terdiri dari perangkat keras (*Hardware)* dan perangkat lunak (*Software).*

3.2.1 Perangkat Keras

Perangkat keras yang digunakan terdiri dari :

1. Notebook Lenovo *Thinkpad* X200 prosesor Intel Centrino RAM 4GB
2. Dua buah *webcam* merk M-tech dengan resolusi 5 Mega Pixel, dengan jarak antar kamera 8 cm dan ketinggian kamera 14 cm
3. Simulasi robot yang terbuat dari kayu dan triplek

3.2.2 Perangkat Lunak

Perangkat lunak terdiri dari Matlab R2017a, di dalam Matlab terdapat Cascade Training yang berfungsi melakukan pengenalan objek serta fungsi *Stereovision* untuk menghitung jarak objek terhadap kamera.

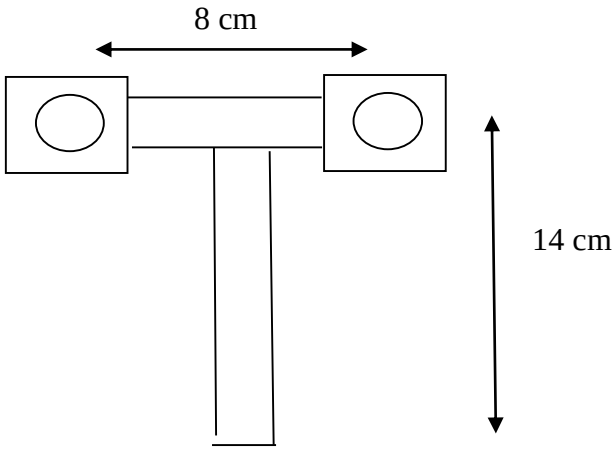
3.3 Perancangan Sistem

Dalam perancangan sistem ini akan dibahas sistem yang dirancang untuk mendeteksi objek sekaligus menghitung jarak terhadap kamera menggunakan metode *Stereovision*, proses perancangan dibagi dalam beberapa tahap yaitu kalibrasi *hardware,* proses pengenalan objek, proses pendeteksian objek, dan proses penghitungan jarak objek terhadap kamera.

Gambar 3.2 Blok diagram sistem

1. *Webcam* merk M-Tech sebanyak dua buah digunakan untuk mengambil gambar objek yang akan dideteksi, sehingga masukan gambar dapat diolah di Notebook dengan perangkat lunak Matlab R2017a dan mengukur jarak objek.
2. Notebook Lenovo *thinkpad* X200 sebagai perangkat keras untuk memproses data dari dua buah webcam yang kemudian diolah menggunakan perangkat lunak Matlab

3. Matlab R2017a sebagai perangkat lunak yang mengolah data yang diterima dari dua buah webcam agar mengenali objek yang ditangkap oleh kamera sekaligus mendeteksi jarak dari kamera dengan menggunakan metode *Stereovision* kemudian ditampilkan hasil nya di Figure Matlab



Gambar 3.3 Rancangan Kamera *Stereo*

Rancangan kamera stereo mengikuti prinsip *stereovison* yaitu penglihatan seperti dua buah mata manusia, sehingga jarak antar kamera diatur sebesar 8 cm atau kurang lebih sama dengan jarak kedua buah bola mata manusia, sedangkan ketinggian sebesar 14 cm menyesuaikan dengan dimensi robot dan arena KRPAI sehingga ketinggian ideal sebesar 14 cm.

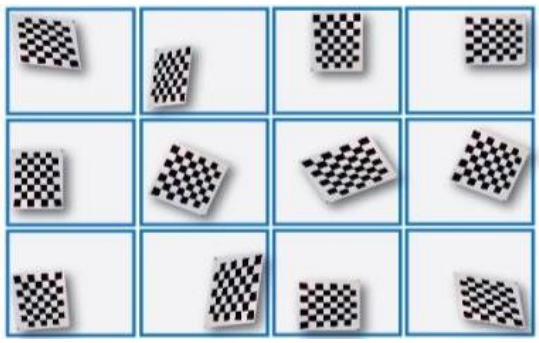
3.3.1 Kalibrasi Perangkat keras

Sebelum menggunakan perangkat keras kamera terlebih dahulu dilakukan kalibrasi dua kamera agar dapat mengakuisisi citra *stereo*, kamera dipersiapkan dengan jarak antar kamera sebesar 8 cm dan ketinggian 14 cm agar dapat dikalibrasi, persiapkan pola kalibrasi (pola kotak-kotak / persegi hitam putih) dengan ukuran kotak yang sama pada permukaan yang rata. Permukaan yang tidak rata akan mempengaruhi keakuratan kalibrasi. Jumlah kotak vertikal dan horizontal tidak boleh sama. satu sisi genap dan sisi lainnya ganjil, misalnya jumlah kotaknya 5x6, tidak boleh 5x5. Pengambilan gambar ini dibantu dengan coding calibration.m

yang telah dipersiapkan untuk mempermudah pengambilan gambar dari kamera 1 dan kamera 2 secara bersamaan. Pada coding ini *delay* pengambilan satu pasang gambar dengan gambar.

Gambar 3.4 Pola catur untuk proses kalibrasi

Ambil sejumlah foto pola kalibrasi dengan stereo kamera yang sudah dipersiapkan, kamera kanan dan kiri harus mengcapture image secara bersamaan atau hampir bersamaan. Untuk hasil kalibrasi terbaik usahakan mengambil 10-20 pasang image, karena nantinya akan ada beberapa pasang *image* yang dibuang dikarenakan gambar yang blur atau pola kalibrasi bergerak saat gambar diambil. Ambil gambar pola kalibrasi dengan jarak dan sudut yang berbeda terhadap stereo

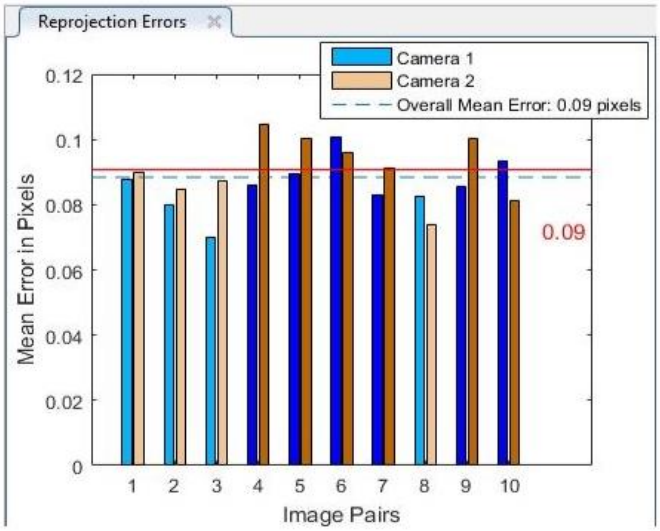


Gambar 3.5 Sudut dan jarak pada pengambilan citra kalibrasi

Dengan menggunakan stereoCameraCalibrator melalui Matlab command prompt dapat melihat hasil evaluasi kalibrasi kamera, dengan cara *load image* dari

kamera 1 dan kamera 2. Masukkan ukuran kotak pola kalibrasi dalam satuan milimeter untuk melakukan proses kalibrasi secara otomatis. Ada kemungkinan beberapa pasang *image* akan ditolak karena blur sehingga pola kotak-kotak tidak dapat dideteksi atau bisa juga karena sudut kemiringan pola kalibrasi yang berlebihan terhadap kamera.

Lakukan evaluasi hasil kalibrasi berdasarkan reproyeksi *error*, apabila ada *error* yang terlalu tinggi pada sebuah pasang image maka hilangkan pasangan citra tersebut sehingga hasil kalibrasi akan lebih akurat. Secara umum reproyeksi *error* yang bagus akan bernilai dibawah 1 pixel. Reproyeksi *error* yang bagus akan menunjukkan hasil seperti berikut ini.



Gambar 3.6 Hasil kalibrasi reproyeksi *error*

Proses terakhir yang harus dilakukan adalah menyimpan dan meng-*export* hasil kalibrasi yang berupa parameter kamera sebagai sebuah obyek Matlab. Pada penelitian ini parameter kamera dinamakan CamParam.

Hasil CamParam ini menyimpan parameter kamera hasil kalibrasi yang terdiri dari :

1.Parameter intrinsik kamera

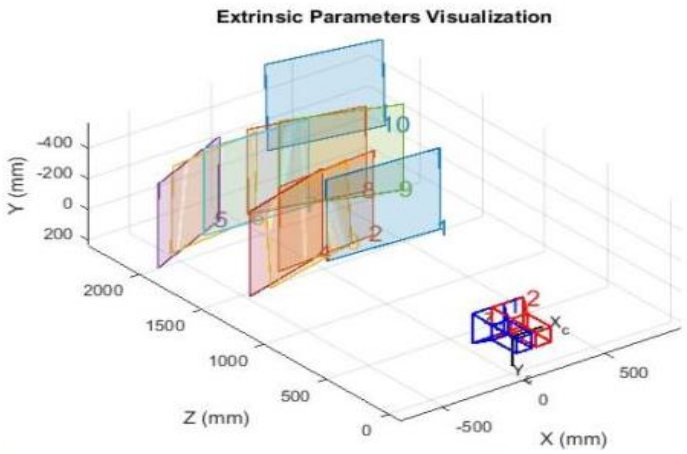
Terdiri dari matriks proyeksi, *optical center*, panjang focal dan kecondongan sumbu kamera

2.Parameter ekstrinsik kamera

Terdiri dari Matriks rotasi 3D, vektor rotasi 3D, dan translasi kamera

3.Distorsi lensa kamera

Terdiri dari koefisien distorsi radial dan koefisien distorsi tangensial

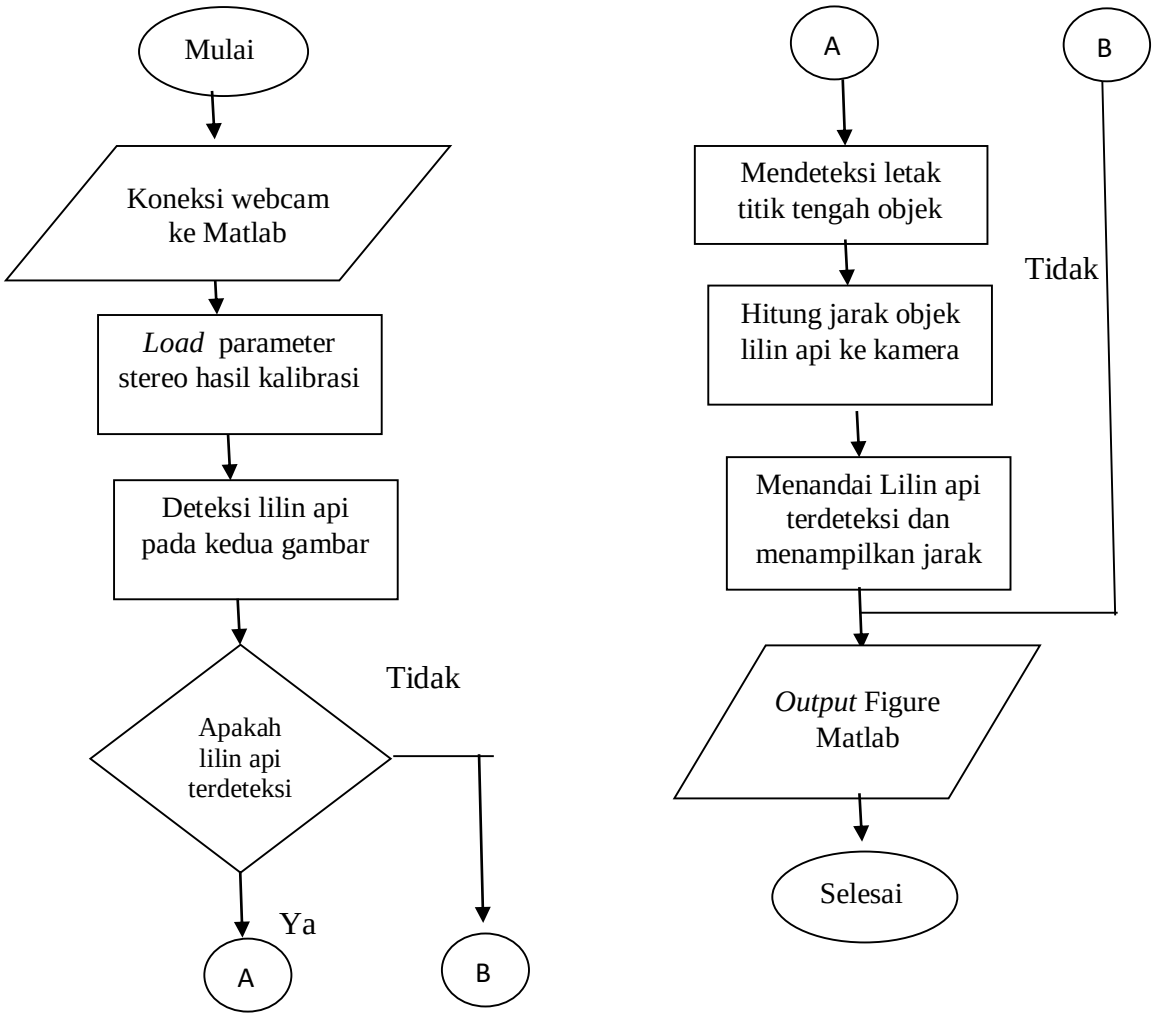


Gambar 3.7 Hasil visualisasi parameter ekstrinsik

Pada gambar 3.7 merupakan hasil visualisasi parameter eksrinsik yang dapat ditampilkan dari hasil pengambilan gambar yang dilakukan untuk kalibrasi, gambar datar berwarna merupakan papan pola kalibrasi yang diletakkan di beberapa posisi untuk proses kalibrasi kamera stereo juga dilengkapi dengan jarak saat melakukan pengambilan gambar kalibrasi.

3.3.2 Perancangan Sistem Perangkat Lunak

Proses perancangan program dilakukan menggunakan aplikasi Matlab R2017a agar mampu mendeteksi objek dan memperkirakan jarak dari kamera, sistem akan mengolah masukan dari kedua buah kamera kemudian mengambil informasi dari parameter objek yang ingin dideteksi sekaligus mengambil informasi titik tengah dari gambar yang berasal dari kedua kamera agar dapat dilakukan perhitungan jarak menggunakan fungsi triangulasi sehingga dapat ditampilkan hasil berupa jarak objek terhadap kamera, untuk tahapan perancangan seperti pada *flowchart* berikut ini:



Gambar 3.8 *Flowchart* Sistem perangkat lunak

1. Koneksi *webcam* ke Matlab

   Proses pertama yang harus dilakukan adalah mengkoneksikan dua buah kamera dengan Matlab, di dalam program menggunakan perintah cam1 = webcam(1); cam2 = webcam(2); sehingga masukan dari kedua buah kamera dapat terbaca di Matlab, juga pengaturan resolusi dengan perintah cam1.Resolution = '320x240'; cam2.Resolution = '320x240'; sehingga kedua kamera dapat diatur resolusinya.

2. *Load* Parameter stereo hasil kalibrasi

   Supaya dapat mengakuisisi citra sesuai dengan metode *stereovision,* maka dilakukan pengunduhan parameter yang telah dihasilkan melalui proses kalibrasi berupa file bernama ‘CamParam.mat’, selanjutnya pada program dilakukan perintah load('CamParam.mat'); untuk mengunduh parameter kamera stereo.

3. Deteksi lilin pada kedua gambar

   Setelah kedua kamera dapat memberikan masukan gambar maka selanjutnya adalah proses deteksi objek lilin api, proses deteksi dapat dilakukan dengan menggunakan perintah LilinDetector = vision.CascadeObjectDetector('Lilin2.xml','UseROI', false); yang terdapat di program adalah mengunduh parameter objek berupa file ‘lilin2.xml’ yang merupakan hasil pelatihan pengenalan objek, sehingga kedua kamera dapat mendeteksi objek lilin api maupun bukan objek lilin api.

4. Mendeteksi titik tengah objek

   Apabila objek terdeteksi kedua kamera, maka proses selanjutnya adalah mengambil informasi titik tengah dari kedua gambar, menggunakan perintah center1 = Lilin1(1:2) + Lilin1(3:4)/2; center2 = Lilin2(1:2) + Lilin2(3:4)/2; diambil informasi titik tengah dari kedua gambar dan selanjutnya dapat diproses ke perhitungan jarak.

5. Perhitungan jarak objek ke kamera

   Perhitungan jarak objek dapat dilakukan setelah mendapatkan informasi titik tengah dari kedua gambar, pada program menggunakan perintah point3d = triangulate(center1, center2, stereoParams); distanceInMeters = norm(point3d)/1000; sehingga dapat diketahui jarak objek dari kedua kamera.

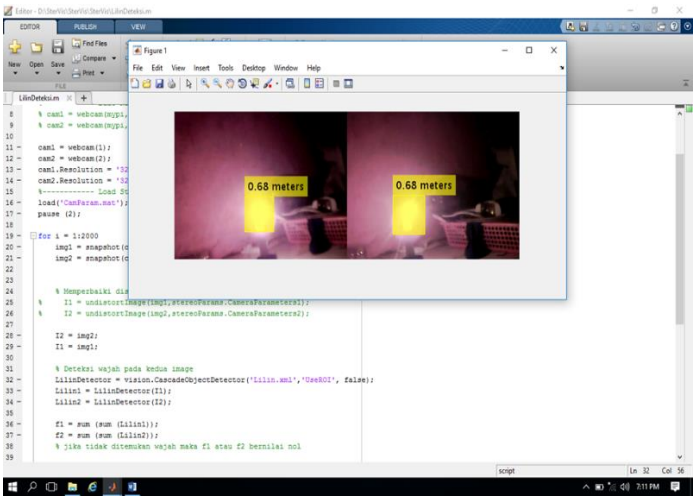
6. Proses Penandaan objek terdeteksi

   Setelah berhasil mendeteksi objek dan menghitung jaraknya maka gambar objek akan ditandai dengan persegi kuning dan juga ditampilkan informasi jaraknya dengan perintah distanceAsString = sprintf('%0.2f meters', distanceInMeters); yaitu menampilkan jarak dalam satuan meter dan perintah:

   I1=insertObjectAnnotation(I1,'rectangle',Lilin1,distanceAsString,'FontSize',18);I2=insertObjectAnnotation(I2,'rectangle',Lilin2,
   distanceAsString,'FontSize',18);
   I1 = insertShape(I1,'FilledRectangle',Lilin1);
   I2 = insertShape(I2,'FilledRectangle',Lilin2); agar memasukan persegi di objek yang terdapat pada kedua buah gambar.

7. *Output* Hasil pendeteksian

   Setelah melakukan penandaan pada objek maka hasilnya akan ditampilkan dalam bentuk figure Matlab yang berisi dua buah gambar dari kedua buah kamera, apabila objek dapat dideteksi maka akan ditampilkan gambar objek dengan penanda persegi pada kedua buah gambar, jika tidak ada objek terdeteksi maka keluaran hanya berupa gambar video dari kedua kamera.



Gambar 3.9 Hasil output pendeteksian jarak

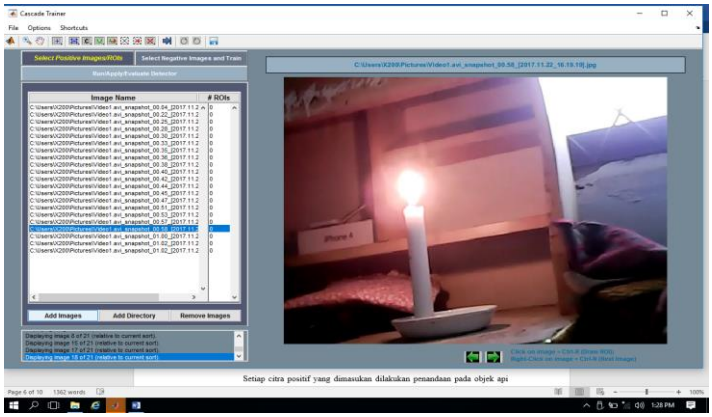
3.3.3 Proses Pengenalan Objek

Proses pengenalan objek dapat dilakukan menggunakan tools Cascade Training yang terdapat di Matlab R2017a, dengan cara mengambil gambar objek yang ingin dilatih dalam bentuk citra positif (*positive image)* dan citra latar belakang (*negative image)* kemudian diolah di tools *Cascade object detector* dengan cara menandai objek yang akan dideteksi kemudian hasil *training* dijadikan parameter dalam sistem yang dibuat untuk mendeteksi objek. Proses pelatihan dibagi dalam beberapa tahap sebagai berikut:

1. *Input* Citra pelatihan

Pada proses ini memasukan citra pelatihan yang berisi objek yang ingin deteksi, dalam proses pelatihan menggunakan Cascade Training, citra pelatihan yang diperlukan minimal sebanyak 20 buah sehingga sistem mampu mengdeteksi objek dengan baik dan semakin banyak jumlah citra pelatihan maka akan semakin meningkatkan akurasi hasil deteksinya, pada proses *input* citra positif kali ini digunakan citra yang diambil sebanyak 30 buah dengan kondisi intensitas cahaya 48 lux, dalam memasukan citra pelatihan juga dimasukan citra yang tidak terdapat objek yang akan digunakan sebagai citra negatif, cascade training akan mendeteksi

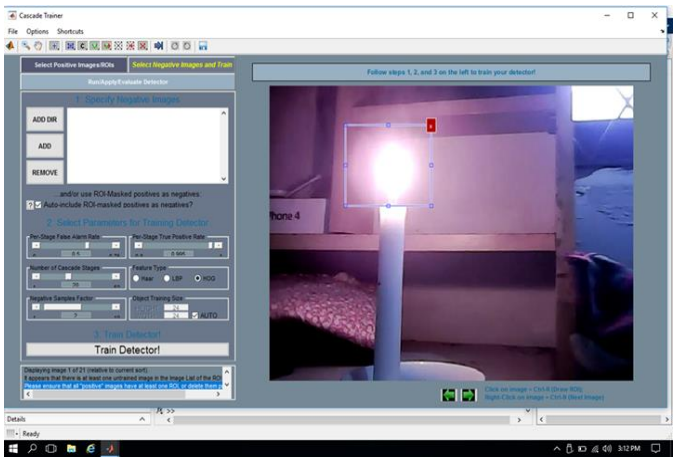
dengan otomatis citra yang tidak memiliki objek api sebagai citra negatif yang akan menjadi citra latar belakang.



Gambar 3.10 Proses *input* citra pelatihan

2. Proses penanda Image ROI

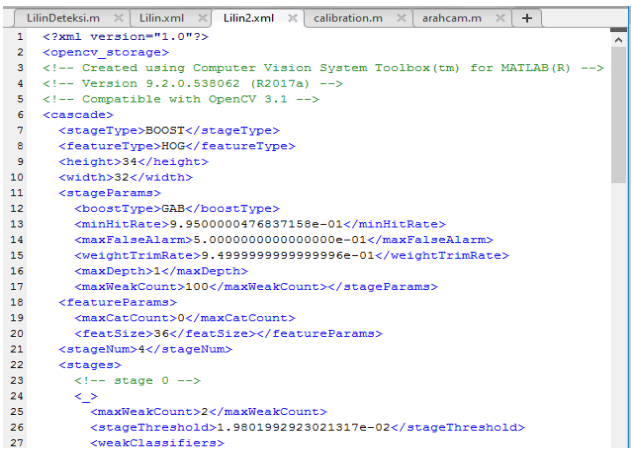
Setiap citra positif yang dimasukan dilakukan penandaan pada objek api yang terdapat di citra pelatihan, proses pembuatan image ROI hanya dengan menandakan objek api pada setiap citra supaya sistem mampu membedakan objek dengan yang bukan objek secara akurat karena parameter objek cukup jelas dengan image ROI pada citra, proses penandaan image ROI cukup dengan menggunakan kursor pada perangkat laptop dan tandai objek membentuk kotak seperti yang terlihat pada gambar 3.9.



Gambar 3.11 Proses penanda dengan image ROI

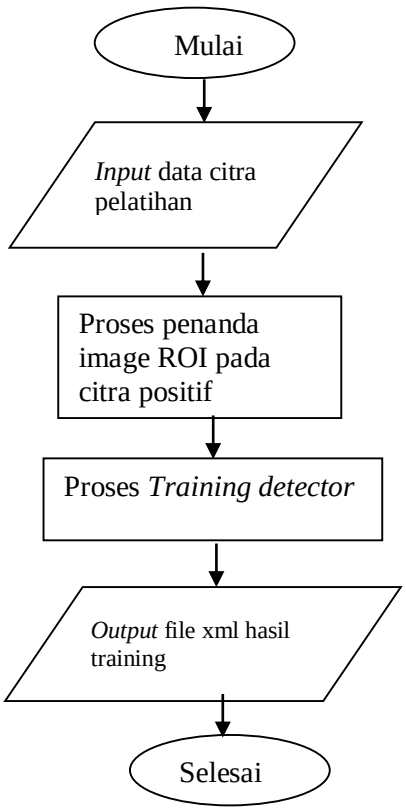
3. Proses *Training* dengan Cascade Training

Dengan menggunakan cascade detector ini dilakukan klasifikasi citra dari image yang dilatih, sistem akan mampu membedakan objek yang latih dengan objek lain yang ada, dataset pelatihan terdiri dari sampel positif dan negatif, sampel data positif adalah data yang berisi citra yang terdapat objek yang ingin dideteksi sedangkan data negatif adalah citra yang tidak terdapat objek yang ingin deteksi dan dapat berupa citra latar belakang saja, pada proses ini menggunakan aplikasi Cascade training yang terdapat di Matlab R2017a, setelah proses pelatihan akan menghasilkan suatu model klasifikasi berupa file xml yang berisi fitur kotak *haar-like*, bobot dan ambang batas tiap klasifikasi, pada folder klasifikasi harus terdapat file xml hasil training ini sebagai parameter objek yang dideteksi sehingga sistem mampu mengenali dan membedakan objek dengan yang bukan objek.



Gambar 3.12 Hasil keluaran file xml

Tahapan proses pelatihan dengan cascade training dijelaskan dalam *flowchart* yang terdapat pada gambar 3.11 berikut ini :



Gambar 3.13 *Flowchart* proses Pelatihan Cascade Training

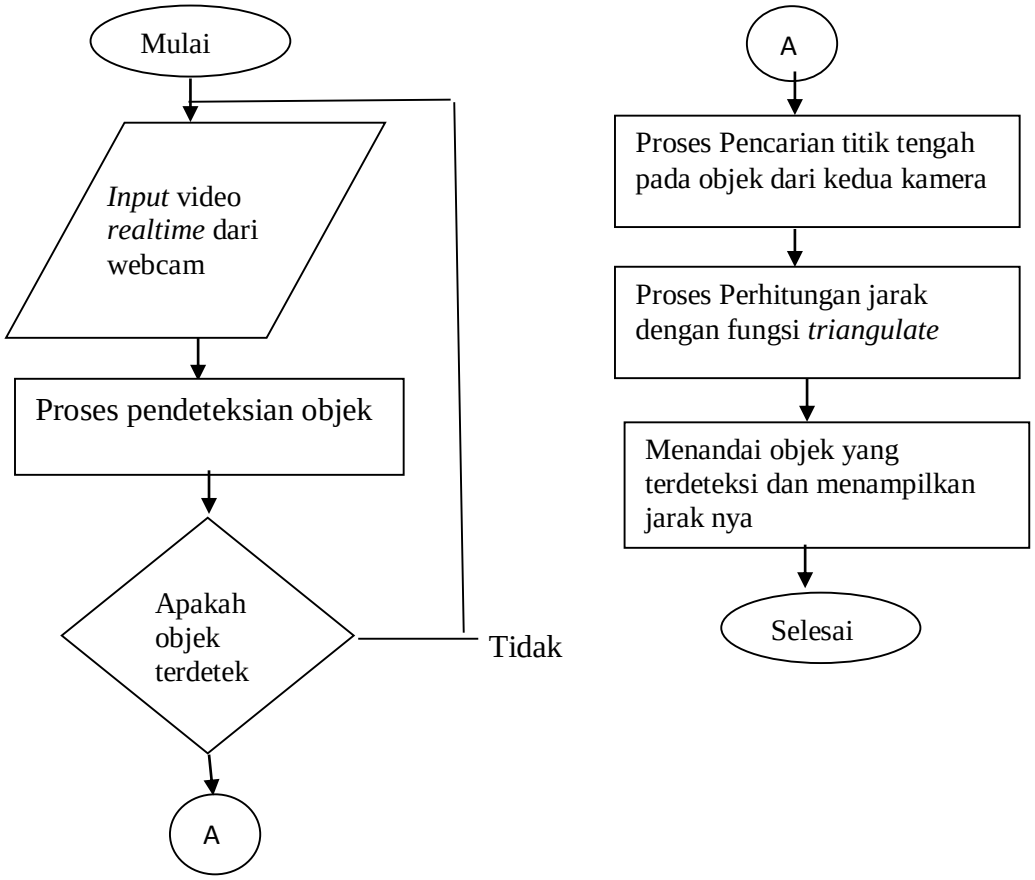
3.3.4 Pendeteksian Objek

Pada proses ini kamera *Stereo* mengambil gambar secara *realtime* dan proses akuisisi gambar dilakukan dengan resolusi 320 x 240, sistem akan mengunduh parameter kamera hasil kalibrasi kamera pada perintah `load('CamParam.mat')` sehingga dapat mengakuisisi citra stereo yang diinginkan kemudian mengunduh parameter hasil pelatihan berupa file xml yang telah ditentukan pada proses pengenalan objek untuk dibandingkan dengan masukan dari kedua kamera, dan selanjutnya objek akan terdeteksi jika sesuai dengan parameter objek yang telah ditentukan menggunakan *Cascade training detector*,proses deteksi objek dilakukan dengan perintah `LilinDetector=vision.CascadeObjectDetector('Lilin.xml','UseROI', false);` yang terdapat pada program , yakni membandingkan hasil masukan citra dari kamera terhadap hasil pelatihan objek dalam bentuk file ‘lilin.xml’ pendeteksian objek juga dipengaruhi oleh kondisi lingkungan dan intensitas cahaya

mempengaruhi keakuratan sistem, maka tingkat akurasi tertinggi jika pendeteksian objek dilakukan di tempat yang memiliki kondisi intensitas cahaya dan *background* yang mirip dengan kondisi pada saat proses *training* objek, kemudian dapat dilakukan pencarian letak titik tengah pada objek yang terdeteksi.

3.3.5 Perhitungan jarak objek

Proses perhitungan jarak dilakukan dengan metode *Stereovision* dengan menggunakan input dari dua buah kamera identik, perhitungan jarak objek dapat dilakukan ketika objek yang dimaksud terdeteksi dan dapat diambil informasi titik tengah dari kedua gambar yang diambil dari kedua buah kamera, nilai titik tengah akan diproses melalui perhitungan menggunakan fungsi triangulasi yang terdapat pada Matlab, pada gambar 3.12 dijelaskan *flowchart* proses pendeteksian dan perhitungan jarak objek :



Gambar 3.14 *Flowchart* proses pendeteksian objek dan perhitungan jarak

Objek yang sudah terdeteksi oleh sistem dapat dilakukan perhitungan jarak terhadap kamera menggunakan informasi titik tengah objek dari kedua kamera yang didapat dengan perintah.

```
center1 = Lilin1(1:2) + Lilin1(3:4)/2;
center2 = Lilin2(1:2) + Lilin2(3:4)/2;
```

Setelah mendapat titik tengah dari kedua kamera dilakukan perhitungan menggunakan fungsi triangulasi yang mampu menghasilkan jarak dari kamera ke pasangan point titik tengah yang terdeteksi sesuai dengan parameter hasil training dalam file ‘ lilin.xml’ , proses perhitungan jarak dilakukan pada perintah `point3d = triangulate(center1, center2, stereoParams)distanceInMeters = norm(point3d)/1000` yang terdapat pada program,

Objek yang berhasil dideteksi akan ditandai dengan persegi warna kuning dan akan ditampilkan juga hasil pengukuran jarak, akurasi perhitungan jarak juga dipengaruhi kondisi lingkungan baik intensitas cahaya maupun latar belakang objek yang dapat berbeda dengan kondisi saat proses *training* objek.

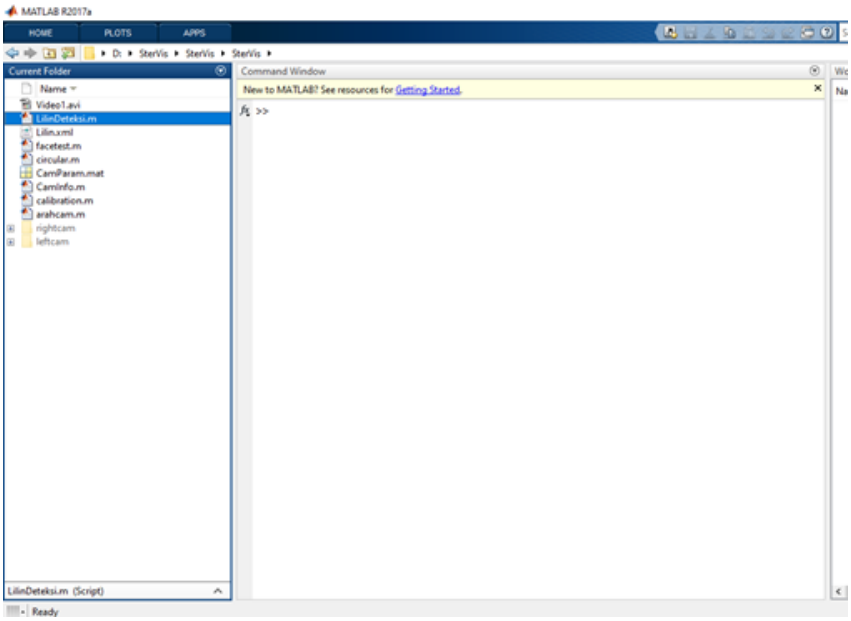
BAB IV
HASIL DAN PEMBAHASAN

4.1 Pengujian Sistem

Pengujian dilakukan menggunakan dua buah *webcam* merk M-tech dengan resolusi 5 Megapixel yang dipasang pada simulasi robot dengan tinggi kamera 14 cm dari permukaan tanah, kedua buah kamera dikoneksikan ke laptop Lenovo dengan kabel USB dan kemudian hasil pengambilan gambar diolah di Matlab R2017a, ketika pengujian posisi perangkat simulasi robot mengambil gambar dari arah pintu masuk tiap ruangan dan mendeteksi objek dengan beberapa variasi jarak.

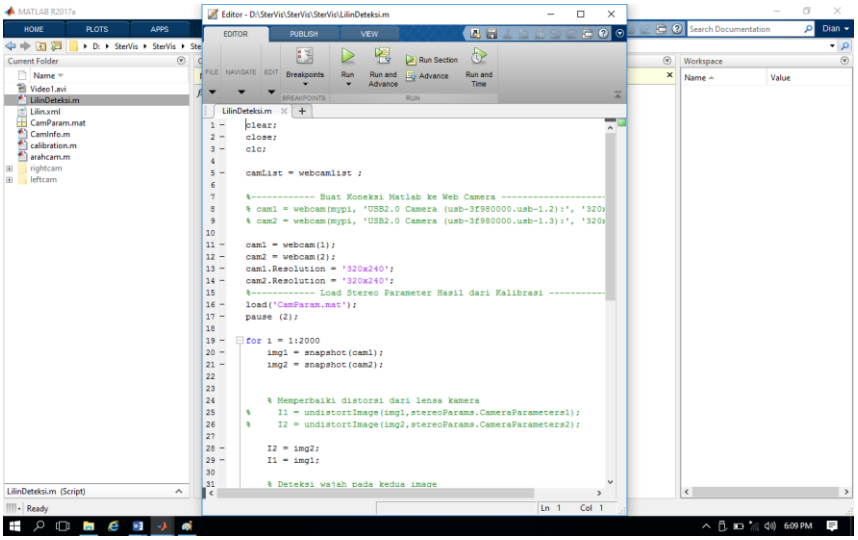
Gambar 4.1 Perangkat keras untuk penelitian

Perangkat keras disiapkan di arena yaitu laptop dan dua buah *webcam* yang sudah dipasang di simulasi robot dan dikoneksikan dengan laptop menggunakan kabel usb 2.0, kedua kamera terpasang dengan jarak 8 cm, dan tinggi kamera dari lantai adalah 14 cm, cara kerjanya dengan membuka aplikasi Matlab R2017a di laptop seperti pada gambar 4.2 berikut.



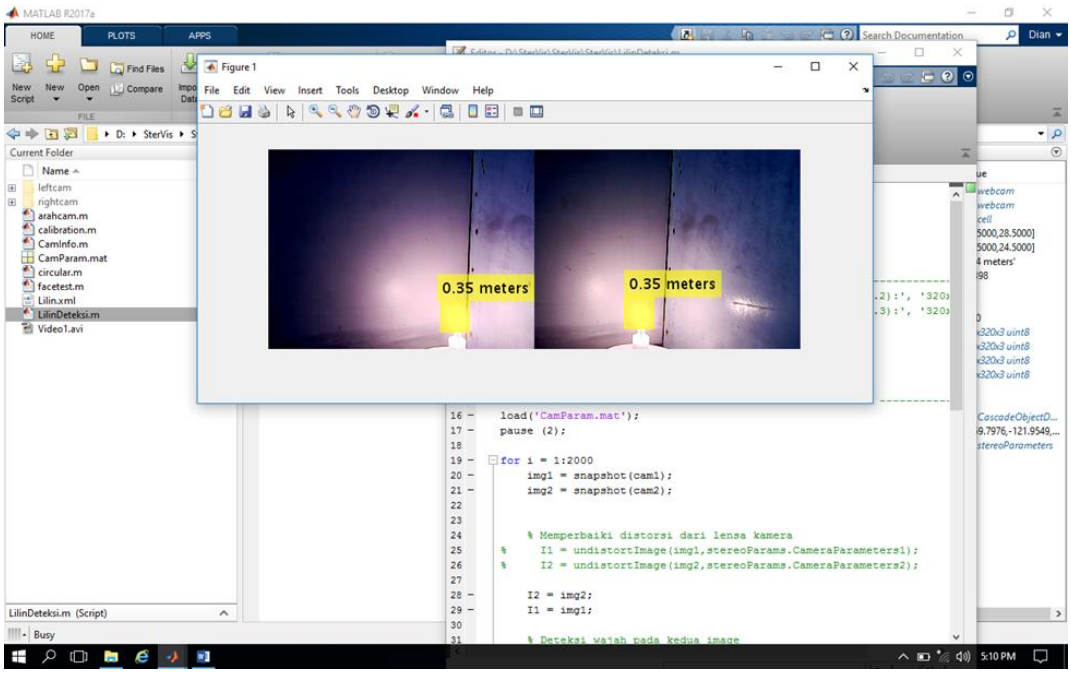
Gambar 4.2 Tampilan awal Matlab R2017a

Pada tampilan awal kita buka folder stervis yang berisi program untuk penelitian dan hasil pelatihan dengan cascade training serta parameter kalibrasi yang keseluruhannya harus diletakkan pada satu folder, lalu berikutnya membuka program Lilindeteksi untuk memulai pengujian dengan tampilan seperti gambar 4.3 berikut.



Gambar 4.3 Tampilan program Lilindeteksi

Setelah program lilindeteksi terbuka seperti gambar 4.3, berikutnya tinggal dijalankan aplikasinya dengan cara menekan tombol F5 atau mengklik tombol Run yang terletak di bagian atas, setelah itu akan muncul tampilan Figure di Matlab dan hasil masukan dari dua buah *webcam* akan terlihat di layar seperti terlihat di gambar 4.6 berikut.



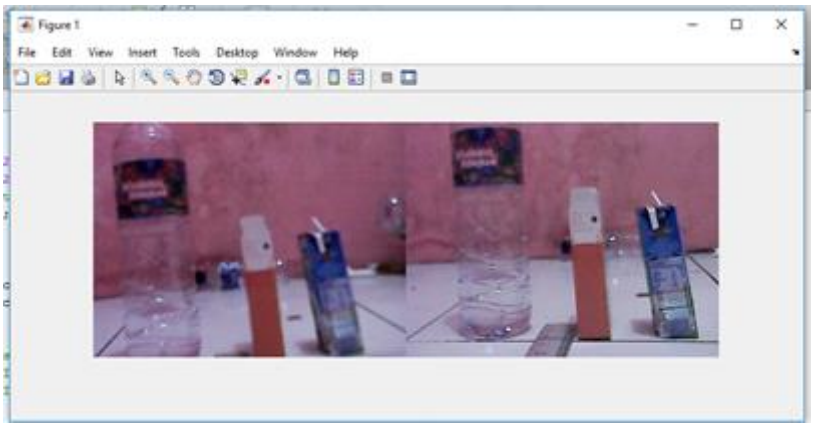
Gambar 4.4 hasil *run* program lilindeteksi

Figure matlab akan menampilkan hasil pengambilan gambar secara *realtime* dari dua buah kamera, apabila kamera menangkap objek lilin api maka akan dideteksi dengan ada persegi kuning sebagai penanda objek dan akan keluar hasil pengukuran jaraknya dengan metode *stereovision* seperti yang terlihat pada gambar 4.4, sehingga program lilindeteksi ini bisa digunakan untuk mendeteksi objek yang tertangkap kamera dan mengukur jaraknya, nilai dari hasil klasifikasi dengan objek yang dideteksi juga didapatkan daengan simbol f1 untuk gambar kamera kiri dan f2 nilai klasifikasi dari gambar kamera kanan, serta nilai titik tengah (center1) dengan nilai (x1,y1) pada gambar kamera kiri dan nilai titik tengah (center2) dengan nilai (x2,y2) dari kamera kanan.

.

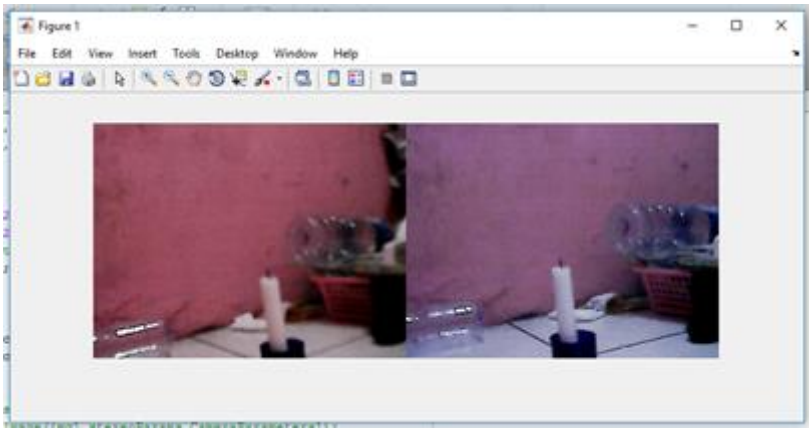
4.1.1 Pengujian dengan objek selain api

Pengujian pertama yang dilakukan adalah menguji keakuratan sistem dalam mengenali objek lilin api dengan objek selain lilin api, sehingga dapat mendeteksi api di arena dengan tepat, pada gambar 4.5 dilakukan pengujian menggunakan beberapa objek.



Gambar 4.5 Pengujian menggunakan objek selain lilin api

Pada gambar 4.5 terlihat hanya gambar hasil masukan dari kedua buah kamera dan tidak mendeteksi objek api,nilai f1 dan f2 yang berasal dari kedua buah gambar bernilai 0 (Lampiran 2.1) atau tidak mendeteksi objek, artinya sistem bekerja dengan cukup baik dan tidak melakukan kesalahan dalam mendeteksi objek.



Gambar 4.6 Pengujian pada objek lilin tidak berapi

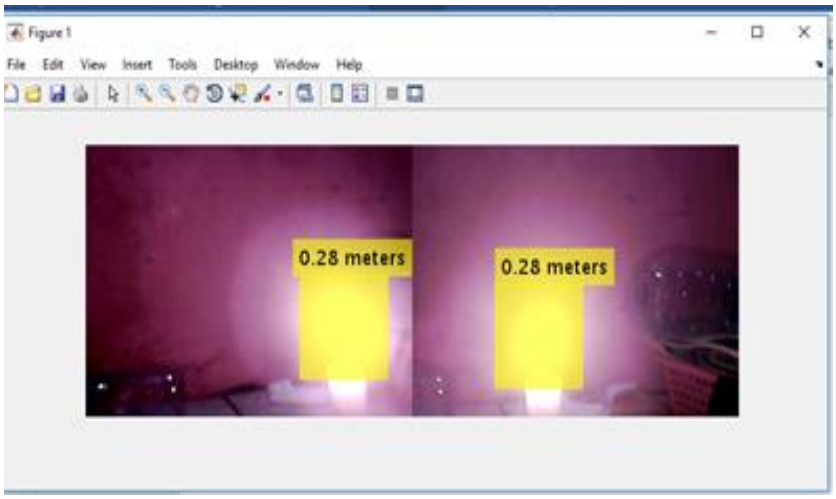
Pada gambar 4.6 dilakukan pengujian pada objek lilin yang tidak ada apinya, nilai f1 dan f2 bernilai nol (Lampiran 2.1.2) sehingga sistem tidak mendeteksi keberadaan api artinya proses pelatihan menggunakan cascade training berhasil mengenali lilin yang memiliki api dan tidak melakukan kesalahan dalam mendeteksi keberadaan api.

4.1.2 Pengujian dengan objek lilin api

Pengujian selanjutnya dengan menggunakan lilin dengan api menyala dan dilakukan beberapa variasi pengujian untuk memastikan keakurasian sistem dalam mendeteksi objek api dan mengukur jarak objek dari kamera, pada pengujian dengan lilin api ini dilakukan pada dua kondisi yaitu kondisi lampu ruangan menyala dan kondisi lampu ruangan mati, serta dilakukan pengujian dengan kondisi lilin api bersama dengan sumber api lain yang berasal dari korek api.

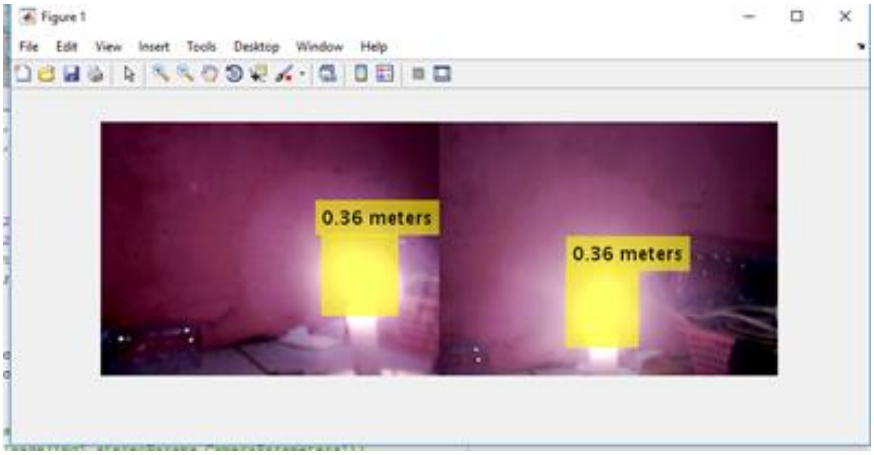
1. Kondisi lampu menyala

   Pada kondisi pertama lampu ruangan menyala dan intensitas cahaya di dalam ruangan sebesar 25 lux, pengujian dilakukan dengan empat variasi jarak, untuk pengujian pertama dilakukan pada jarak 28 cm.



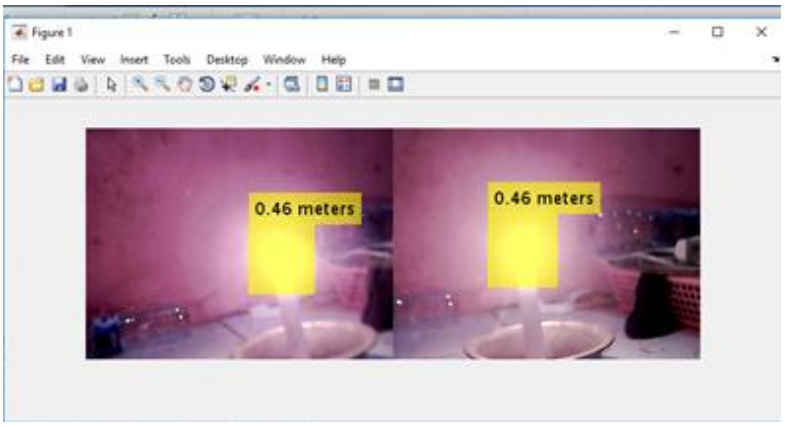
Gambar 4.7 Pengujian pada jarak 28 cm.

Pada jarak 28 cm, sistem dapat mendeteksi api dengan akurat dengan nila f1 dari gambar kamera kiri sebesar 472 dan nilai f2 sebesar 710, informasi titik tengah dari kamera kiri (center1) sebesar (217.50, 165) dan nilai center2 sebesar (190.50 , 79), nilai titik tengah diproses ke fungsi perhitungan *Stereovision* sehingga dapat mengukur jarak dengan akurat sebesar 28 cm seperti terlihat pada gambar 4.7, selanjutnya dilakukan pengujian dengan jarak 36 cm.



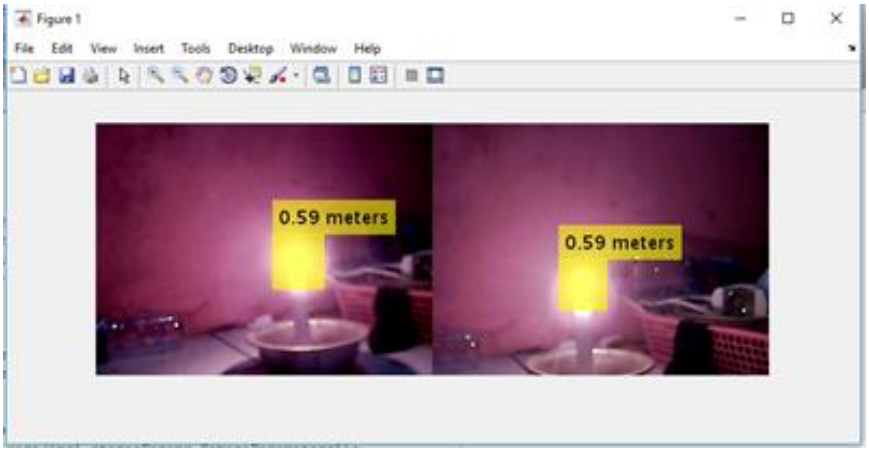
Gambar 4.8 Pengujian pada jarak 36 cm

Pada jarak 36 cm sistem dapat medeteksi objek dengan nilai f1= 412 dan nilai f2=343 serta titik tengah center1 = (164.50 , 183) dan center2= (135,89.50), sistem mampu mengukur jarak dengan cukup akurat seperti terlihat pada gambar 4.8, selanjutnya dilakukan pada jarak 46 cm.



Gambar 4.9 Pengujian pada jarak 46 cm.

Pada gambar 4.9 terlihat hasil pengukuran sebesar 46 cm, artinya sistem dapat mengenali objek secara tepat dengan nilai f1=450 dan f2=370 serta nilai titik tengah center1=(193.50 , 185) dan center2=(154.50 , 122) dari informasi titik tengah dapat mengukur jarak secara akurat, pengujian selanjutnya dilakukan pada jarak 60 cm.



Gambar 4.10 Pengujian pada jarak 60 cm.

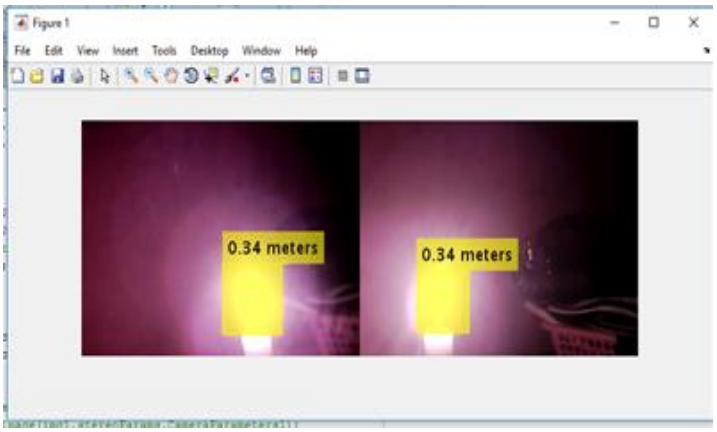
Pada jarak 60 cm sistem mendeteksi objek dengan tepat dengan nilai f1=427 dan f2=317 dan hasil titik tengah center1=(222.50 , 100) dan center2=(179.50 , 72,50), pada hasil pengukuran ada selisih jarak sebesar 1 cm, pada gambar 4.10 terlihat hasil pengukuran sebesar 59 cm, dari seluruh pengujian pada kondisi intensitas cahaya 25 lux ini berhasil mengenali objek pada empat variasi jarak dengan hasil pengujian keseluruhan pada tabel 4.1 berikut ini.

Tabel 4.1 Hasil Pengujian pada kondisi intensitas cahaya 25 lux

Jarak *real (Cm)*	**Jarak terukur (Cm)**	**Selisih Jarak (Cm)**	**% Error**
28	28	0	0
36	36	0	0
48	48	0	0
60	59	1	1,66
% Rata-rata Error			**0,415**

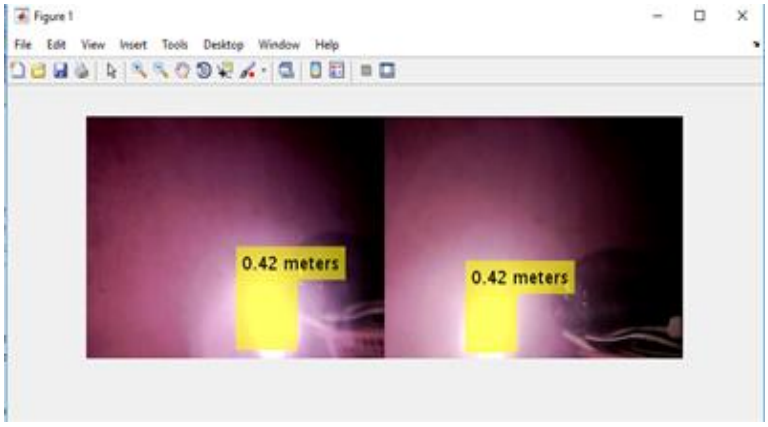
2. Pengujian pada kondisi lampu tidak menyala

   Pengujian berikutnya dilakukan pada kondisi ruangan gelap tanpa lampu menyala dengan intensitas cahaya terbaca sebesar 7 lux, pengujian dilakukan dengan empat variasi jarak, pengujian pertama dilakukan pada jarak 30 cm.



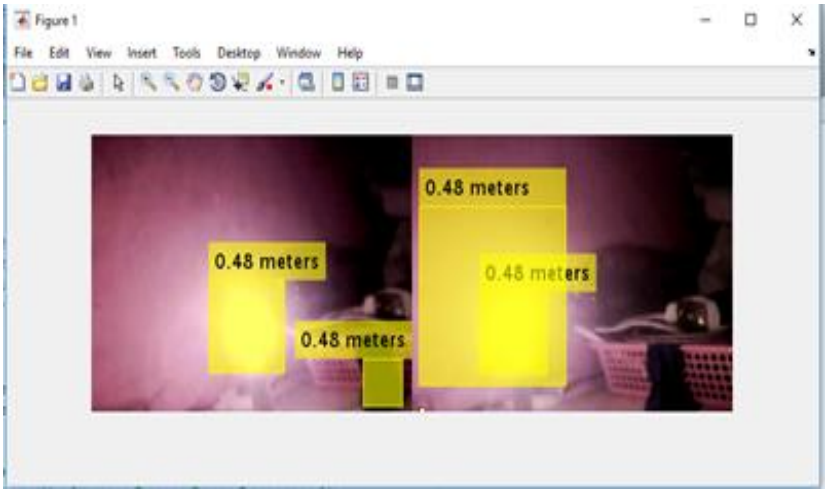
Gambar 4.11 Pengujian pada jarak 30 cm.

Pada jarak 30 cm sistem dapat mendeteksi objek dengan akurat dengan nilai pendeteksian f1=422 dan f2=296 serta nilai titik tengah center1= (211.50 , 106.50) dan center2=(106 , 103.50) seperti terlampir di lampiran 2.3.1, namun pada gambar 4.11 hasil pengukuran sebesar 34 cm, ada selisih pengukuran dengan jarak aslinya sebesar 4 cm, untuk pengukuran berikutnya dilakukan pada jarak 40 cm.



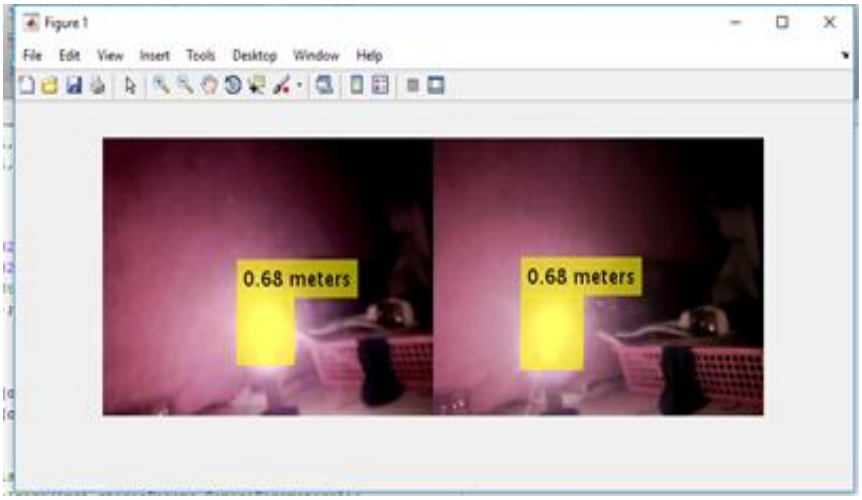
Gambar 4.12 Pengujian pada jarak 40 cm

Pada jarak 40 cm sistem dapat mendeteksi objek dengan akurat, nilai f1 terbaca sebesar 362 dan nilai f2 sebesar 207, nilai titik tengah center1 sebesar (170.50 , 106) dan center2 sebesar (99.50, 107) seperti terlampir di lampiran 2.3.2, namun terdapat selisih jarak sebesar 2 cm, pada gambar 4.12 jarak terukur sebesar 42 cm sehingga terdapat selisih jarak sebesar 2 cm, untuk pengukuran berikutnya dilakukan pada jarak 50 cm.



Gambar 4.13 Pengujian pada jarak 50 cm.

Pada gambar 4.13, objek berhasil dideteksi (nilai f1=356, f2=306) hasil pengukuran membaca sebesar 48 cm, dengan informasi titik tengah center1 (175.50, 107) dan center2 = (104, 117.50) seperti terlampir di lampiran 2.3.3, ada selisih dengan jarak aslinya yaitu 50 cm dengan selisih jarak sebesar 2 cm, pengujian selanjutnya dilakukan pada jarak 60 cm.



Gambar 4.14 Pengujian pada jarak 60 cm.

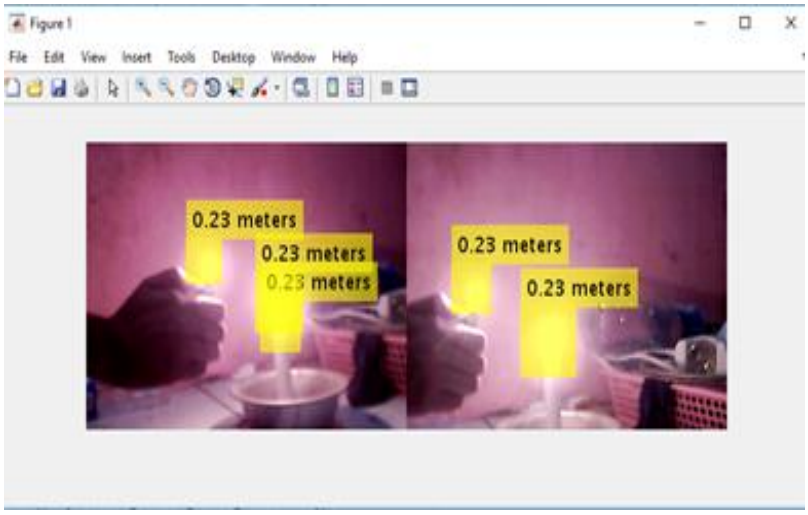
Pada jarak 60 cm, sistem dapat terdeteksi dengan nilai f1= 304 dan f2= 293 (lampiran 2.3.4), selisih pengukuran cukup besar karena sistem membaca jarak sebesar 68 cm seperti terlihat pada gambar 4.14 dari informasi titik tengah center1 = (179.50,110) dan center2= (102,120.50) seperti terlampir di lampiran 2.3.4, hal ini dapat disebabkan minimnya cahaya sehingga sistem gagal mengukur jarak dengan akurat.

Tabel 4.2 Hasil Pengujian pada kondisi intensitas cahaya 7 lux

Jarak *real* (Cm)	**Jarak terukur (Cm)**	**Selisih Jarak**	**% Error**
30	34	4	13,33
40	42	2	5
50	48	2	4
60	68	8	13,33
% Rata-rata Error			**8,91**

3. Pengujian menggunakan dua buah api

   Pada pengujian ini menggunakan lilin api namun ada objek api lain yang berasal dari korek api seperti pada gambar 4.15 berikut ini:



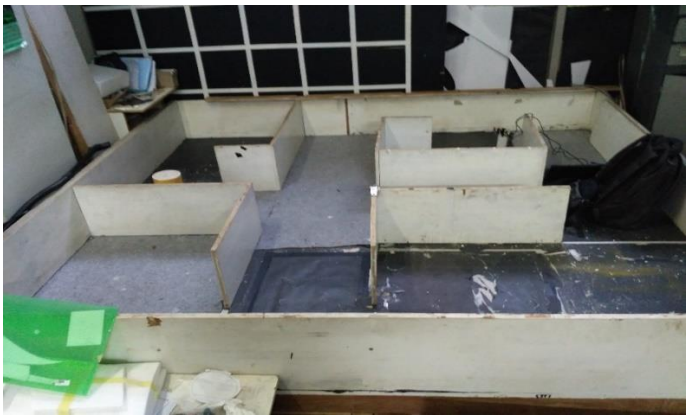
Gambar 4.15 Pengujian dengan dua buah api

Pada pengujian ini kedua objek api dapat terdeteksi oleh sistem, hal ini disebabkan objek lilin api yang digunakan untuk pelatihan memiliki bentuk api yang mirip dengan korek api sehingga kedua objek dapat dikenali, pengujian dilakukan dengan jarak 23 cm dan kondisi intensitas cahaya 25 lux pada kondisi ruangan dengan lampu menyala, kedua objek dapat diukur jaraknya dengan cukup akurat seperti terlihat pada gambar 4.15 dengan besaran nilai f1 yang terbaca sebesar 553 dan f2 sebesar 884, serta nilai tengah center1 sebesar (109,170.50) dan center2 sebesar (42,43) seperti terlampir di lampiran 2.4, hal ini bisa disebabkan sistem mendeteksi nilai f1 dan f2 dari kedua buah objek api serta informasi titik tengah berasal dari kedua objek.

Secara keseluruhan sistem berjalan baik di pengujian sistem ini, objek terdeteksi pada kondisi nilai f1 dan f2 memiliki rentan nilai di atas 300 maka objek dapat dideteksi sebagai api berdasarkan parameter dari hasil pelatihan menggunakan Cascade Training, serta kondisi penerangan cahaya maupun pengambilan gambar dengan kamera mempengaruhi keakuratan jarak karena nilai titik tengah bisa berbeda saat kondisi cahaya dari api berubah dan ketidaksesuaian dengan kondisi saat pelatihan juga mempengaruhi keakurasian dalam mendeteksi objek dan mengukur jarak.

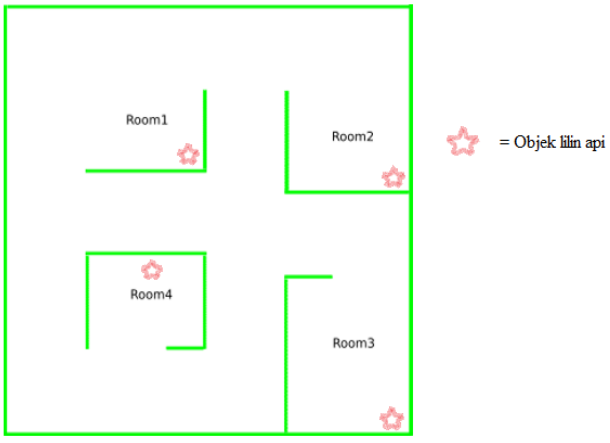
4.2 Pengujian di arena KRPAI

Pengukuran jarak objek dilakukan di arena KRPAI yang terdiri dari empat ruangan sesuai dengan rules KRPAI 2017, di tiap ruangan diletakkan lilin api dan kamera akan mendeteksi objek api sekaligus mengukur jaraknya



Gambar 4.16 arena KRPAI

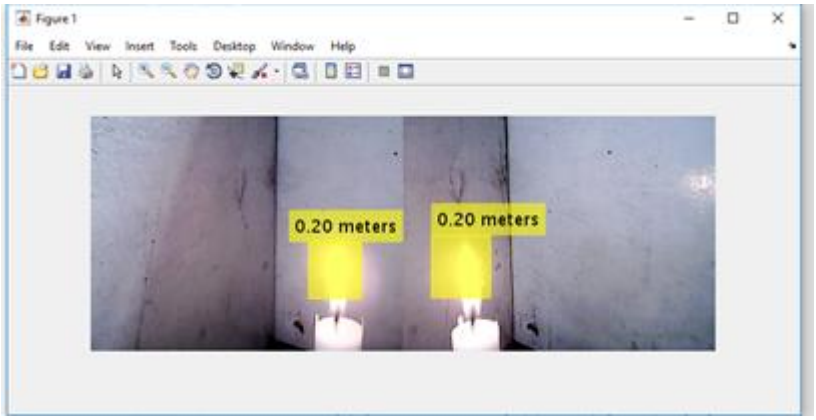
Dilakukan pengujian untuk mendeteksi api di tiap sudut ruangan yang tersedia, pengujian dilakukan menggunakan lilin api dan dilakukan dalam berbagai variasi jarak di setiap ruangan arena KRPAI dengan jangkauan terdekat sejauh 20 cm.



Gambar 4.17 Lokasi objek lilin api di arena

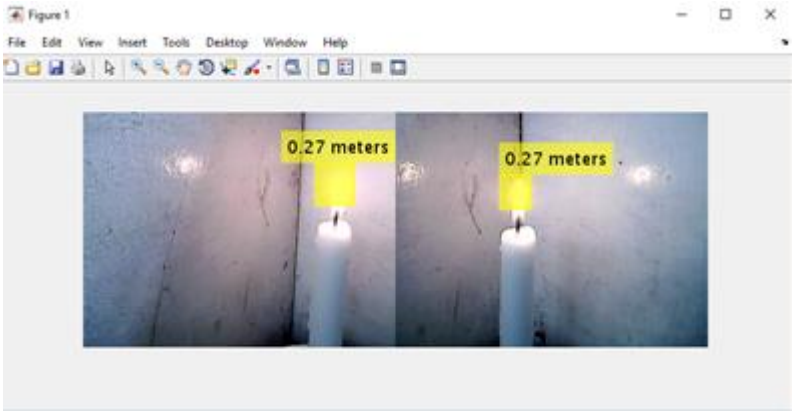
4.1.1 Pengujian di Ruangan 1

Pengujian pertama dilakukan di ruangan 1 dengan kondisi intensitas cahaya sebesar 580 lux, di ruangan ini dilakukan pengukuran sebanyak empat kali dengan jarak berbeda, dengan posisi kamera setinggi 14 cm maka jarak terdekat yang dipakai dalam pengujian ini adalah 20 cm.



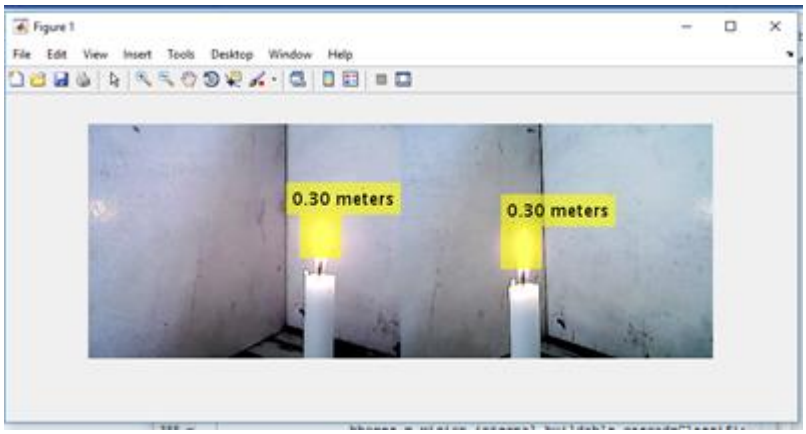
Gambar 4.18 pengukuran pada jarak 20cm di ruangan 1

Terlihat di Gambar 4.18 pengujian sistem mendeteksi api di ruangan 1 pada jarak 20 cm berhasil dengan baik, api mampu dideteksi dan hasil pengukuran nya dapat ditampilkan di layar figure pada Matlab,nilai titik tengah yang terdeteksi pada gambar kamera kiri (center1) dengan nilai (256.50, 120) dan nilai center2 dari kamera kanan sebesar (211,127.50) seperti terlampir di lampiran 2.5.1, dengan hasil pengukuran yang sesuai dengan jarak sesungguhnya lilin api dan kamera.



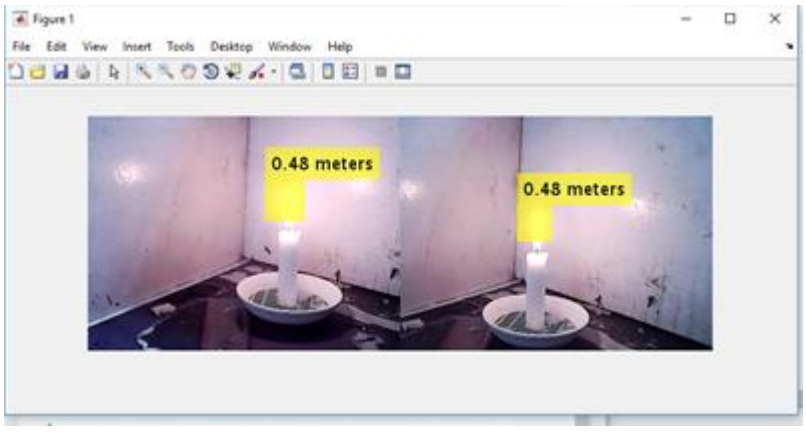
Gambar 4.19 pengukuran jarak 27 cm di ruangan 1

Kemudian dilakukan juga pengujian pada jarak 27 cm, dan sistem masih dapat mendeteksi objek dengan cukup baik seperti terlihat pada gambar 4.19, sekaligus menampilkan hasil pengukuran yang cukup akurat sesuai dengan jarak sesungguhnya,nilai titik tengah terdeteksi dengan center1 sebesar (214.50, 135) dan center2 sebesar (135.50, 189) seperti terlampir di lampiran 2.5.2, berikutnya dilakukan kembali pengukuran pada jarak 30 cm.



Gambar 4.20 pengukuran pada jarak 30 cm di ruangan 1

Pada jarak 30 cm, sistem masih dapat mendeteksi dengan cukup baik objek api dan berhasil mengukur jarak dengan akurat seperti terlihat pada gambar 4.20, nilai center1 sebesar (209, 118.50) dan center2 sebesar (128.50, 114) seperti terlampir di lampiran 2.5.3, .dalam tiga kali pengukuran didapatkan hasil yang cukup akurat, kemudian dilakukan kembali pengukuran pada jarak 40 cm.



Gambar 4.21 pengukuran pada jarak 40 cm di ruangan 1

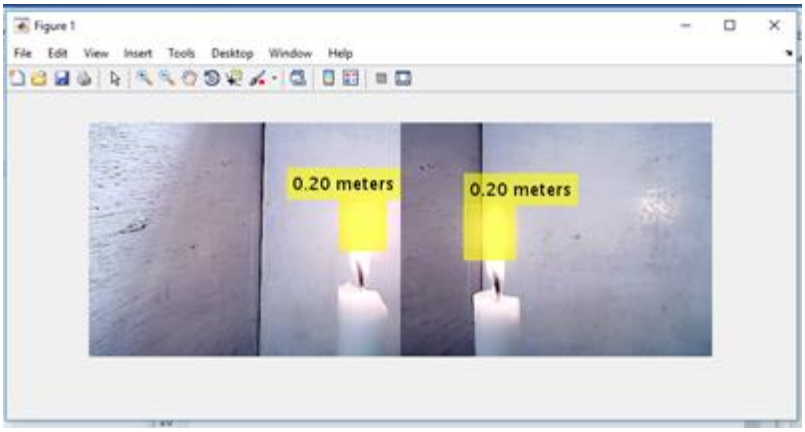
Pada jarak 40 cm, objek masih dapat terdeteksi namun jarak yang terukur sebesar 48 cm seperti terlihat di gambar 4.21, artinya terdapat selisih jarak dengan jarak aslinya sebesar 8 cm, nilai center1 sebesar (255, 108.50) dan center2 sebesar (137, 119) seperti terlampir di lampiran 2.5.4, hal ini disebabkan keterbatasan sistem dalam mengenali objek api yang dilakukan pada pengenalan objek dengan cascade detector sehingga ada ketidakakuratan dalam pengukuran jarak.

Tabel 4.3 hasil pengukuran di ruangan 1

Jarak *real* (cm)	**Hasil pengukuran (cm)**	**Selisih hasil (cm)**	**% Error**
20	20	0	0
27	27	0	0
30	30	0	0
40	48	8	20
Rata-rata % error			5

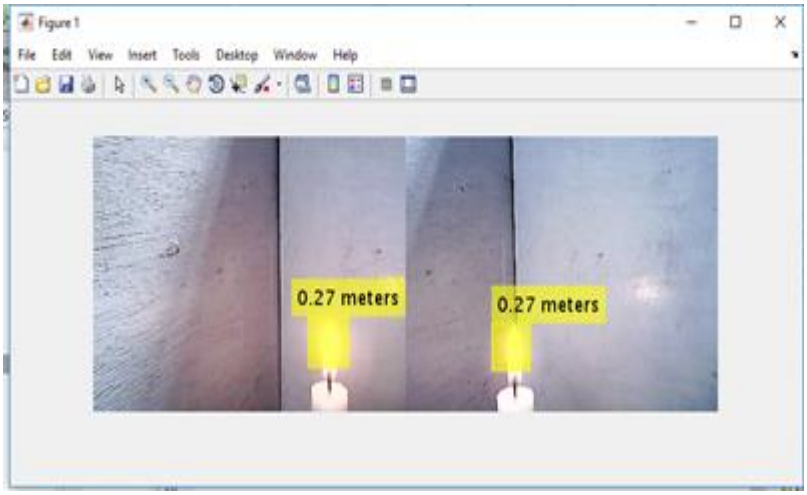
4.1.2 Pengujian di ruangan 2

Pengujian selanjutnya dilakukan di ruangan 2 dengan lilin api ditaruh di sudut ruangan 2, dengan kondisi intensitas cahaya sebesar 236 lux dilakukan pengujian sebanyak 4 kali dengan jarak yang berbeda, pada jarak terdekat dilakukan pengujian pada jarak 20 cm.



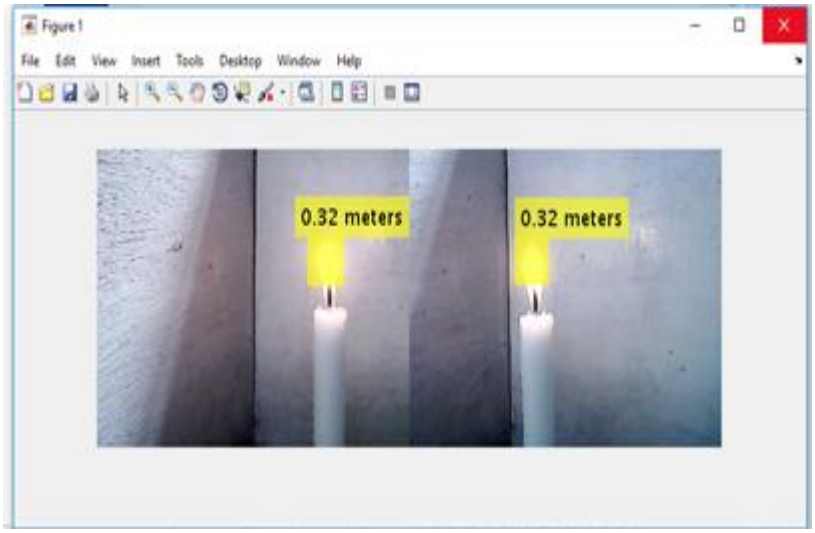
Gambar 4.22 pengukuran pada jarak 20 cm di ruangan 2

Pada jarak 20 cm, sistem dapat mendeteksi dengan cukup baik dan dapat mengukur jarak dengan akurat seperti pada gambar 4.22, informasi titik tengah center1 sebesar (218.50, 172) dan center2 sebesar (155.50, 177.50) seperti terlampir di lampiran 2.6.1, kemudian dilakukan kembali pengukuran dengan jarak 27 cm.



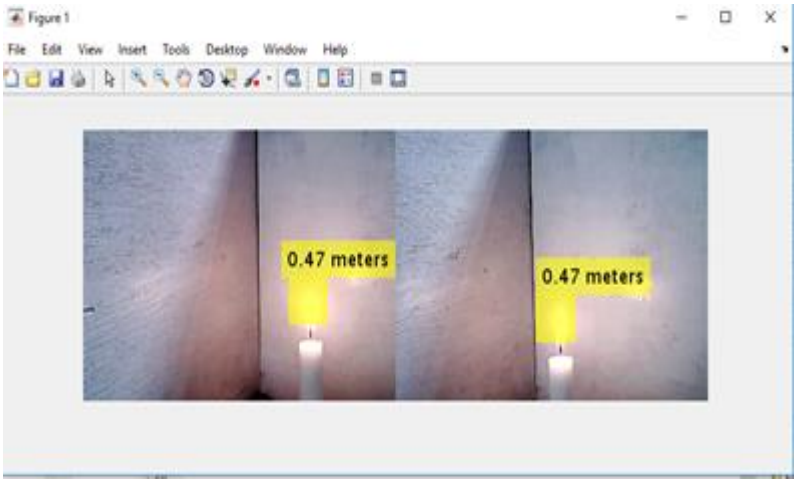
Gambar 4.23 pengukuran pada jarak 27 cm di ruangan 2

Pada gambar 4.23, sistem dapat mendeteksi objek dengan baik dan mampu mengukur jarak secara akurat pada jarak 27 cm, sesuai dengan jarak aslinya, informasi titik tengah center1 sebesar (230,177.50) dan center2 sebesar (80.50, 177.50) seperti terlampir di lampiran 2.6.2.



Gambar 4.24 Pengukuran pada jarak 30 cm di ruangan 2

Pengujian pada jarak 30 cm sistem dapat mendeteksi objek dengan baik seperti terlihat di gambar 4.24 namun ada selisih jarak pengukuran dengan jarak aslinya sebesar 2 cm, informasi titik tengah center1 sebesar (258,182.50) dan center2 sebesar (155.50, 172) seperti terlampir di lampiran 2.6.3.



Gambar 4.25 Pengukuran pada jarak 40 cm di ruangan 2

Pada jarak 40 cm, sistem masih mendeteksi objek dengan baik namun hasil pengukuran sebesar 47 cm seperti pada gambar 4.25, informasi titik tengah center1 sebesar (168,272) dan center2 sebesar (85.50, 179) seperti terlampir di lampiran

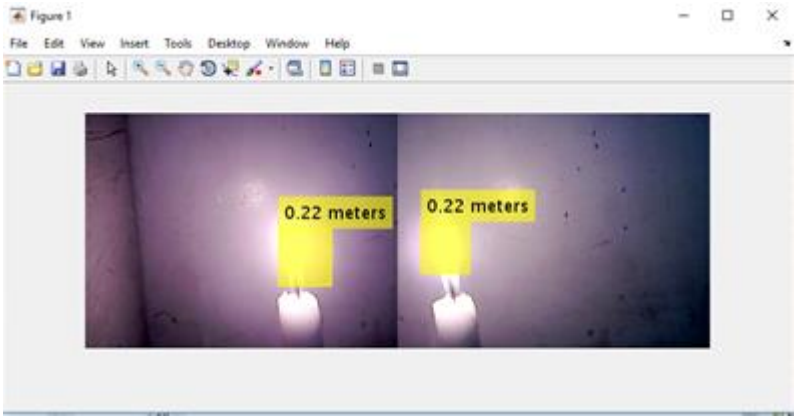
2.6.4, ada selisih jarak sebesar 7 cm pada jarak ini, seperti pada ruangan 1 pada jarak yang semakin jauh ada kesalahan pengukuran.

Tabel 4.4 hasil pengukuran di ruangan 2

Jarak *real* (cm)	**Hasil pengukuran (cm)**	**Selisih jarak (cm)**	**% Error**
20	20	0	0
27	27	0	0
30	32	2	6,66
40	47	7	17,5
Rata-rata % Error			6,04

4.1.3 Pengujian di ruangan 3

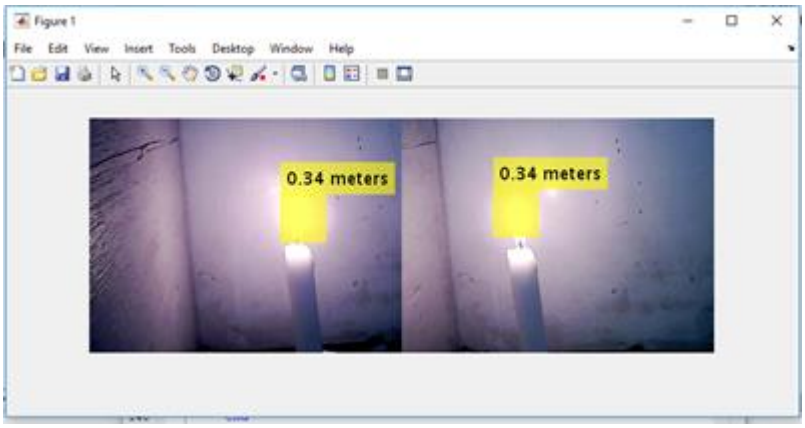
Pada pengujian selanjutnya dilakukan di ruangan 3 sesuai denah arena KRPAI dengan intensitas cahaya sebesar 90 lux, lilin api diletakkan di pojok ruangan dan sistem mendeteksi objek nya.



Gambar 4.26 Pengukuran pada jarak 20cm di ruangan 3

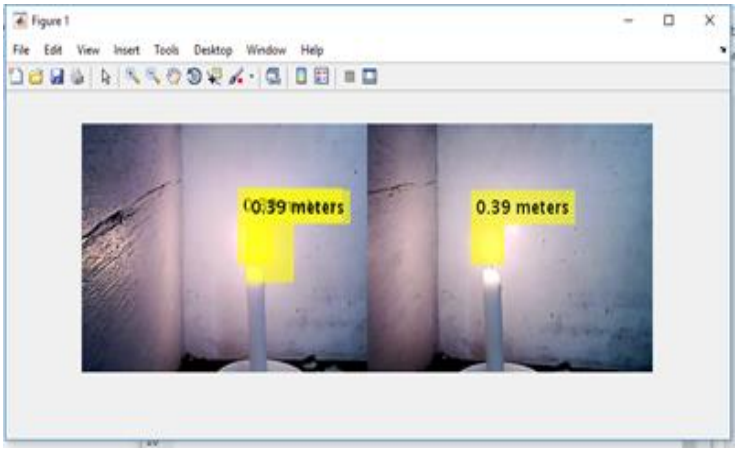
Pada pengukuran pertama di jarak 20 cm terjadi perbedaan jarak terukur dengan jarak aslinya seperti terlihat pada gambar 4.26, di layar figure matlab

didapatkan hasil terukur sebesar 22 cm, ada selisih jarak sekitar 2 cm, dipengaruhi oleh informasi titik tengah yang ditemukan dengan nilai center1 sebesar (260,182.50) dan center2 sebesar (58.50, 172) seperti terlampir di lampiran 2.7.1.



Gambar 4.27 Pengukuran pada jarak 30cm di ruangan 3

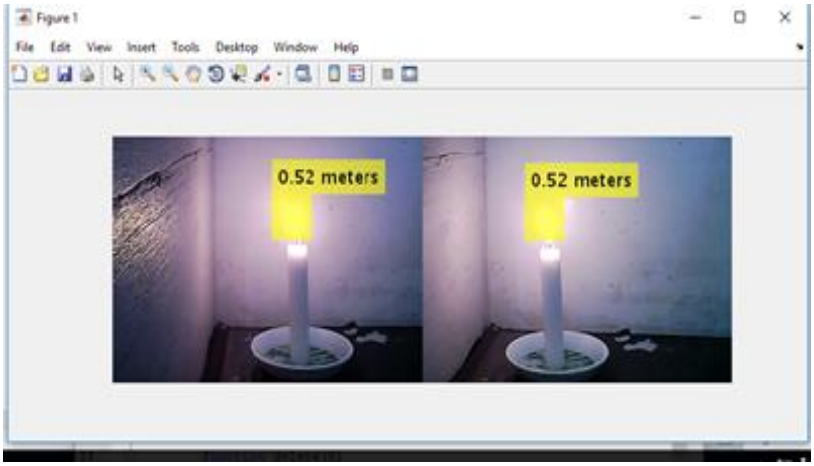
Terlihat di gambar 4.27, pada jarak 30 cm sistem dapat mengenali objek api dengan baik namun sistem membaca jarak sebesar 34 cm sehingga terjadi selisih pengukuran sebesar 4 cm, selisih jarak disebabkan nilai titik tengah yang terbaca dengan nilai center1 sebesar (205, 108.50) dan center2 sebesar (127.50, 116) seperti terlampir di lampiran 2.7.2.



Gambar 4.28 Pengukuran pada jarak 40cm di ruangan 3

Pada jarak 40 cm, sistem dapat mengenali objek dengan baik dan jarak dapat terukur sebesar 39 cm seperti terlihat pada gambar 4.28, besaran titik tengah center1

yang terdeteksi sebesar (218.5, 103) dan titik tengah center2 sebesar (131.50 , 121.50) seperti terlampir di lampiran 2.7.3, dari hasil pengukuran ada selisih 1 cm dari jarak aslinya yang bisa disebabkan ketidaktelitian dalam pengukuran jarak asli ataupun kondisi intensitas cahaya di ruangan.



Gambar 4.29 Pengukuran pada jarak 50cm di ruangan 3

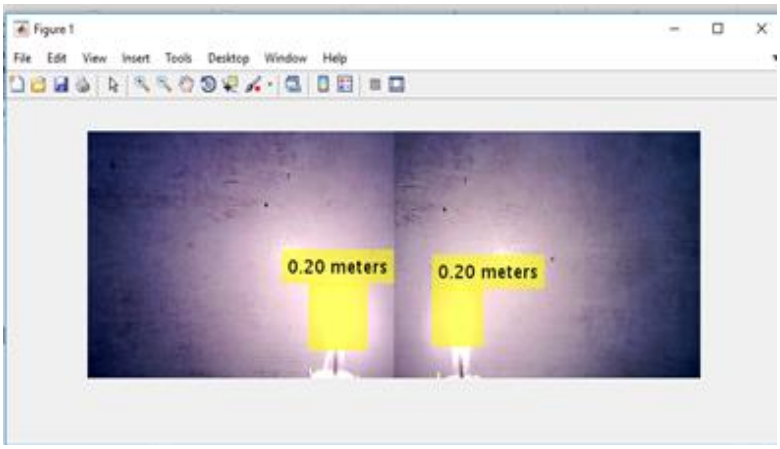
Pada jarak 50 cm, sistem berhasil mengenali objek dan mengukur jarak dengan hasil sebesar 52cm seperti terlihat pada gambar 4.29, sehingga ada selisih dengan jarak asli sebesar 2 cm, hasil pengukuran diperoleh dari perhitungan menggunakan nilai titik tengah center1 sebesar (253.50, 118) dan center2 sebesar (125.50, 115) seperti terlampir di lampiran 2.7.4.

Tabel 4.5 Hasil pengujian di ruangan 3

Jarak *real* (cm)	**Jarak terukur (cm)**	**Selisih Jarak (cm)**	**% Error**
20	22	2	10
30	34	4	13,33
40	39	1	2,5
50	52	2	4
Rata – rata % Error			**7,45 %**

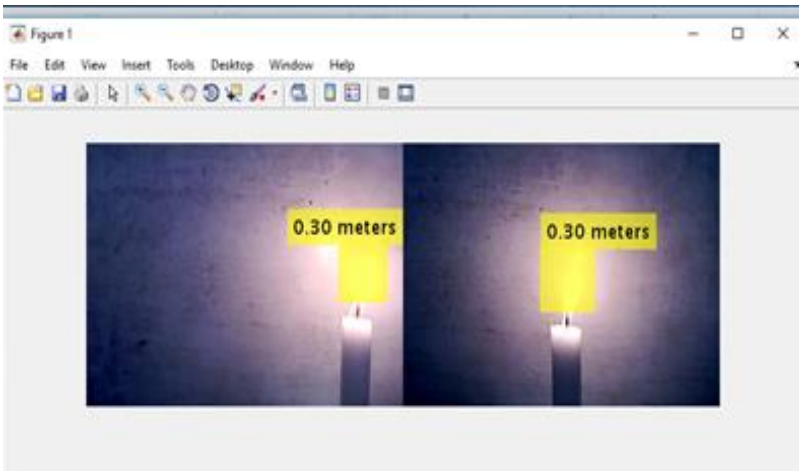
4.1.4 Pengujian di Ruangan 4

Pada pengujian berikutnya dilakukan di ruangan 4, dengan kondisi intensitas cahaya 133 lux, objek lilin api diletakkan di sudut ruangan dan pengujian dilakukan dari jarak terdekat 20 cm.



Gambar 4.30 Pengukuran pada jarak 20 cm di ruangan 4

Terlihat di gambar 4.30 pada pengukuran di jarak 20 cm, sistem dapat mengenali objek dengan baik dan dapat mengukur jarak dengan akurat sebesar 20 cm sesuai dengan jarak aslinya, pengukuran jarak didapatkan dari informasi titik tengah center1 sebesar (238,172) dan center2 sebesar (55.50, 177.50) seperti terlampir di lampiran 2.8.1, selanjutnya dilakukan pengukuran pada jarak 30 cm.

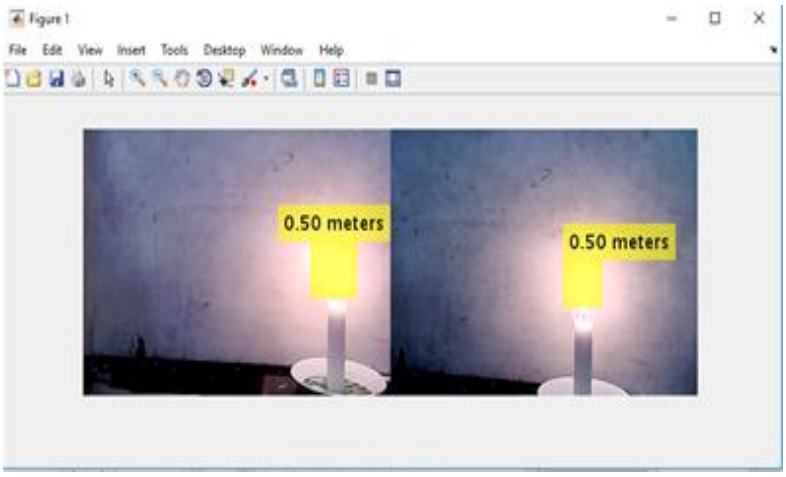


Gambar 4.31 Pengukuran pada jarak 30cm di ruangan 4

Pada jarak 30 cm, sistem pun dapat mengenali objek dan dapat mengukur jarak sebesar 30 cm seperti terlihat pada gambar 4.31, nilai titik tengah yang terdeteksi center1 dengan nilai (281,122.50) dan center2 (142.50,116) seperti terlampir di lampiran 2.8.2, artinya sistem dapat mengukur jarak dengan akurat sesuai dengan jarak aslinya.

Gambar 4.32 Pengukuran pada jarak 40 cm di ruangan 4

Pada jarak 40 cm, sistem dapat mengenali objek dan mampu mengukur jarak dengan hasil 41 cm seperti terlihat pada gambar 4.32, terdapat selisih dengan jarak aslinya sebesar 1 cm, nilai titik tengah center1 sebesar (175.50, 106) dan center2 sebesar (99.50, 117) seperti terlampir di lampiran 2.8.3.



Gambar 4.33 Pengukuran pada jarak 50cm di ruangan 4

Pada gambar 4.33, dengan jarak 50 cm sistem dapat mengenali objek dan mengukur jarak objek dengan akurat yaitu 50 cm, informasi titik tengah center1 (177.50, 107) dan center2 = (108, 127.50) seperti terlampir di lampiran 2.8.4.

Tabel 4.6 Hasil pengujian di ruangan 4

Jarak *real* (cm)	**Jarak terukur (cm)**	**Selisih Jarak (cm)**	**% Error**
20	20	0	0
30	30	0	0
40	41	1	2,5
50	50	0	0
Rata-rata % Error			**0,625**

Pengujian di ruangan 4 memiliki rata-rata kesalahan paling kecil yaitu sebesar 0,625%, pada pengujian ini sistem mendeteksi dan mengukur jarak di empat jarak berbeda dan hanya mengalami kesalahan di pengukuran pada jarak 40 cm dengan selisih jarak sebesar 1 cm.

4.3 Analisa hasil pengujian

Pengujian telah dilakukan di arena Kontes Robot Pemadam Api Indonesia (KRPAI) sesuai rules Trinity College Fire-fighting Home Robot Contest 2016 dengan konfigurasi ruangan level A, tiap ruangan memiliki kondisi intensitas cahaya yang berbeda beda sehingga mempengaruhi hasil pengujian.

Tabel 4.7 Hasil pengujian di arena KRPAI

Room	**Jarak *real* (cm)**	**Jarak terukur (cm)**
1	20	20
	27	27
	30	30
	40	48
	Rata-rata error	**5 %**
2	20	20
	27	27
	30	32
	40	47
	Rata-rata Error	**6,04 %**

Room	**Jarak *real* (cm)**	**Jarak terukur (cm)**
3	20	22
	30	34
	40	39
	50	52
	Rata-rata error	**7,45 %**
4	20	20
	30	30
	40	41
	50	50
	Rata-rata Error	**0,625 %**
% Error total		**4,77 %**

Setelah dilakukan pengujian dengan hasil seperti di tabel 4.5 dapat dianalisa bahwa secara keseluruhan sistem dapat bekerja dengan baik dengan rata-rata eror keseluruhan sebesar 4,77 %, dimana rentang *error* terendah terdapat di ruangan 4 dengan *error* sebesar 0,625% sedangkan *error* tertinggi terdapat di ruangan 3 dengan *error* sebesar 7,45%, dapat disimpulkan bahwa sistem bekerja dengan cukup baik di ruangan 4 dimana kondisi intensitas cahaya sebesar 133 lux, pada pengukuran jarak di ruangan 4 hanya pengukuran pada jarak 40 cm terdapat selisih jarak sebesar 1 cm, selebihnya sistem dapat mengukur jarak dengan akurat, sedangkan pada ruangan 3 yang memiliki *error* terbesar dengan kondisi intensitas cahaya sebesar 90 lux sistem bekerja kurang optimal, hal ini dapat disebabkan oleh kondisi cahaya yang kurang baik di ruangan tersebut sehingga kamera tidak menangkap gambar objek terlalu baik dan kurang dapat menghitung jarak dengan

akurat yang disebabkan kurang akuratnya nilai titik tengah yang didapatkan, faktor lain yang dapat mempengaruhi adalah kurang baiknya kondisi *hardware* saat melakukan pengujian maupun pemasangan perangkat keras yang kurang sesuai sehingga mempengaruhi hasil pengujian, jika dibandingkan dengan hasil penelitian dari referensi [2] pada objek lilin dengan rata-rata total kesalahan sebesar 3,88% atau lebih baik dibanding hasil pengujian penulis, hal ini dikarenakan perbedaan dalam penggunaan perangkat keras kamera yang digunakan oleh penelitian dari referensi [2] dan metode pengambilan gambarnya juga berbeda karena pada penelitian referensi [2] mengambil gambar secara bergantian dari kamera kiri dan kamera kanan sedangkan pada penelitian penulis mengambil gambar bersamaan secara *realtime*, dan latar belakang objek juga berbeda jika di penelitian referensi [2] menggunakan kain hitam agar bisa membedakan objek dengan warna cerah sedangkan penulis melakukan pengujian di arena Kontes Robot Pemadam Api Indonesia (KRPAI) dengan latar belakang objek berupa tembok arena berwarna putih, sehingga hasil akurasinya berbeda.

BAB V
PENUTUP

5.1 Kesimpulan

Dari hasil penelitian yang telah dilakukan didapatkan beberapa kesimpulan sebagai berikut :

1. Pengenalan objek dengan Cascade Training pada aplikasi MATLAB berhasil dilakukan dengan baik, sistem dapat mengenali objek dengan akurat hasil akuisisi citra menggunakan dua buah *webcam* pada kondisi lingkungan dan intensitas cahaya yang sesuai sehingga dapat menghitung jarak objek terhadap kamera.
2. Metode *Stereovision* untuk mengukur jarak suatu objek dari kamera di arena Kontes Robot Pemadam Api Indonesia (KRPAI) dapat berjalan dengan cukup baik menggunakan perhitungan triangulasi dari titik tengah objek yang berada di dua kamera, dengan hasil ukur yang cukup akurat dengan persentase rata-rata kesalahan sebesar 4,77%.

5.2 Saran

1. Perlu penggunaan perangkat keras yang lebih baik untuk hasil pengambilan citra yang lebih baik .
2. Dalam melakukan pelatihan objek agar dapat menggunakan citra yang lebih banyak demi meningkatkan akurasi hasil.

DAFTAR PUSTAKA

[1] Pramudyo,Anggoro. Febrian,Reza. dan Wiryadinata,Romi. "*Deteksi Objek pada Arena Kontes Robot Api Indonesia Menggunakan Raspberry Pi dan OpenCV*"Seminar Nasional Expo Teknik Elektro 2015 , Cilegon.

[2] Muhimmah,Izati. Novian,Mahardika. Meilita. Rahmalianto,Deni. Dhomas Hatta,Fudholi "*Metode Stereovision untuk memperkirakan jarak objek dari kamera*" Universitas Islam Indonesia 2012, Sleman.

[3] Salim,David. Suriani. Andianto "*Pengukuran Jarak berbasis Stereo Eyes"* Bina Nusantara University Jakarta 2009.

[4] Ferry,Stefanus Widiatama,Setiawan. Willian,Surya " *Studi Eksperimen dalam perhitungan kedalaman objek 3D dengan pendekatan Stereo vision* " Bina Nusantara University Jakarta 2010

[5] Chandra, Eka Putra, Tommy Budiman, Iman Herwidiana Kartowisastro "*Estimasi Posisi Objek Berdasarkan Stereovision System"* Bina Nusantara University Jakarta 2013

[6] Jernej Mrovlje, Damir Vrancic "*Distance Measuring based on stereoscopic pictures*" 9th International PhD workshop on systems and control 1-3 October 2008, Slovenia.

[7] Ioannis Kostavelis Lazaros Nalpantidis , Antonios Gasteratos "*Real-Time Algorithm for Obstacle Avoidance using a stereoscopic camera*" Democritus University of Thrace,University Campus , Kimmeria 2008 , Greece.

[8] Chandra,Devy. dan Prajna,Gaja. Nagarjuna dan Agung Nugroho , Lintang "*Studi Pendeteksian Wajah dengan Metode Viola-Jones* " Bina Nusantara University Jakarta 2011

[9] Helvig Jensen "*Implementation the Viola-Jones Face detection Algorithm* " Technical University of Denmark Kongen Lyngby 2008

[10] Permana,Indra. M. Zen. Samsono,Hadi. Setiawardhana "*Tracking Object menggunakan metode template matching berbasis Stereovision*" Politeknik Elektronika Negeri Surabaya Institut Teknologi Sepuluh November Surabaya 2013

LAMPIRAN

Listing program lilindeteksi

```
clear;
close;
clc;

camList = webcamlist ;

%------------ Buat Koneksi Matlab ke Web Camera ----------------------

cam1 = webcam(1);
cam2 = webcam(2);
cam1.Resolution = '320x240';
cam2.Resolution = '320x240';
%------------ Load Stereo Parameter Hasil dari Kalibrasi --------------
load('CamParam.mat');
pause (2);

for i = 1:2000
    img1 = snapshot(cam1);
    img2 = snapshot(cam2);


    % Memperbaiki distorsi dari lensa kamera
%     I1 = undistortImage(img1,stereoParams.CameraParameters1);
%     I2 = undistortImage(img2,stereoParams.CameraParameters2);

    I2 = img2;
    I1 = img1;

    % Deteksi lilin pada kedua image
    LilinDetector = vision.CascadeObjectDetector('Lilin.xml','UseROI',
false);
    Lilin1 = LilinDetector(I1);
    Lilin2 = LilinDetector(I2);

    f1 = sum (sum (Lilin1));
    f2 = sum (sum (Lilin2));
    % jika tidak ditemukan lilin maka f1 atau f2 bernilai nol

    % jika ditemukan lilin maka hitung jaraknya ke kamera
    if f1 == 0 || f2 == 0
        imshowpair(I1, I2, 'montage');
    else
        % Temukan letak titik pusat lilin
        center1 = Lilin1(1:2) + Lilin1(3:4)/2;
        center2 = Lilin2(1:2) + Lilin2(3:4)/2;

        % Hitung jarak lilin ke kamera
```

```
        % fungsi triangulate menghasilkan jarak/lokasi dari kamera ke
pasangan point
        % pada kedua stereo image berdasarkan stereo parameter hasil
kalibrasi
        point3d = triangulate(center1, center2, stereoParams);
        distanceInMeters = norm(point3d)/1000;

        % Menandai lilin yang terdeteksi dan menampilkan jaraknya
        distanceAsString = sprintf('%0.2f meters', distanceInMeters);
        I1 =
insertObjectAnnotation(I1,'rectangle',Lilin1,distanceAsString,'FontSize',18);
        I2 = insertObjectAnnotation(I2,'rectangle',Lilin2,
distanceAsString,'FontSize',18);
        I1 = insertShape(I1,'FilledRectangle',Lilin1);
        I2 = insertShape(I2,'FilledRectangle',Lilin2);

        imshowpair(I1, I2, 'montage');
     end
end
```

listing kalibrasi

```
clear;
clc;

for i = 1:15
    for n = 1:10
        imgle = snapshot(cam1);
        imgri = snapshot(cam2);
        imshow (imgle);
        drawnow
    end
    beep
    pause (3)
    imgl = snapshot(cam1);
    imgr = snapshot(cam2);
    ii = string (i);
    stringl = ["left",ii,".jpg"];
    stringr = ["right",ii,".jpg"];
    strl = join(stringl);
    strr = join(stringr);
    namel = char(strl);
    namer = char(strr);
    imwrite(imgl,namel);
    imwrite(imgr,namer);
    pause (3);
end
```

listing caminfo

```
clear;
close all;
clc;
```

```
%%
%%
%-------------- Buat Koneksi Matlab ke Web Camera ----------------------
cam1 = webcam(mypi, 'USB2.0 Camera (usb-3f980000.usb-1.2):', '320x240');
cam2 = webcam(mypi, 'USB2.0 Camera (usb-3f980000.usb-1.3):', '320x240');

%%
%-------------- Load Stereo Parameter Hasil dari Kalibrasi --------------
load('CamParam.mat');

%%
% Visualkan parameter kamera ekstrinsik hasil kalibrasi
showExtrinsics(stereoParams);

pause (2);
beep
pause (3);
%%
%-------------- Capture Image dari web camera ----------------------
img1 = snapshot(cam1);
img2 = snapshot(cam2);



%%
%------------ Rectify (Ralat) kedua gambar dari masing2 webcam -----------
% sehingga point yang sesuai terletak pada baris  piksel yang sama pada kedua
image
[frameLeftRect, frameRightRect] = rectifyStereoImages(img1, img2,
stereoParams);

% Tampilkan stereo anaglyph image hasil penggabungan kedua rectified image
figure;
imshow(stereoAnaglyph(frameLeftRect, frameRightRect));
title('Stereo Anaglyph Image');

%%
%-------------------- Perhitungan Disparity Map ----------------------
% disparity merupakan jarak antar piksel yang sesuai pada kedua rectified
image
% disparity proporsional dengan jarak kamera ke point pada keadaan sebenarnya
frameLeftGray  = rgb2gray(frameLeftRect);
frameRightGray = rgb2gray(frameRightRect);
disparityMap = disparity(frameLeftGray, frameRightGray);

% Tampilkan disparity map
figure;
imshow(disparityMap, [0, 64]);
title('Disparity Map');

%%
%-------------------- Rekonstruksi 3D Scene ----------------------

points3D = reconstructScene(disparityMap, stereoParams);

% Konversi ke meter dan buat pointCloud object
```

```
points3D = points3D ./ 1000;
ptCloud = pointCloud(points3D, 'Color', frameLeftRect);

% Buat streaming point cloud viewer
player3D = pcplayer([-3, 3], [-3, 3], [0, 8], 'VerticalAxis', 'y', ...
    'VerticalAxisDir', 'down');
%%
% Visualisasikan point cloud
view(player3D, ptCloud);
colormap jet
colorbar
```

Hasil pelatihan lilin.xml

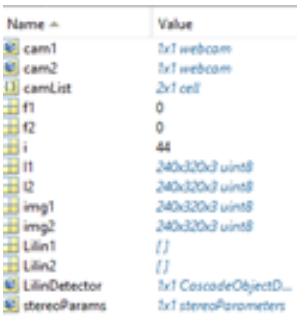
```
<?xml version="1.0"?>
<opencv_storage>
<!-- Created using Computer Vision System Toolbox(tm) for MATLAB(R) -->
<!-- Version 9.2.0.538062 (R2017a) -->
<!-- Compatible with OpenCV 3.1 -->
<cascade>
  <stageType>BOOST</stageType>
  <featureType>HOG</featureType>
  <height>34</height>
  <width>32</width>
  <stageParams>
    <boostType>GAB</boostType>
    <minHitRate>9.9500000476837158e-01</minHitRate>
    <maxFalseAlarm>5.0000000000000000e-01</maxFalseAlarm>
    <weightTrimRate>9.4999999999999996e-01</weightTrimRate>
    <maxDepth>1</maxDepth>
    <maxWeakCount>100</maxWeakCount></stageParams>
  <featureParams>
    <maxCatCount>0</maxCatCount>
    <featSize>36</featSize></featureParams>
  <stageNum>4</stageNum>
  <stages>
    <!-- stage 0 -->
    <_>
      <maxWeakCount>2</maxWeakCount>
      <stageThreshold>1.9801992923021317e-02</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 3 9.8355099558830261e-02</internalNodes>
          <leafValues>
            -9.7979795932769775e-01 1.</leafValues></_>
        <_>
          <internalNodes>
            0 -1 6 8.3104997873306274e-02</internalNodes>
          <leafValues>
```

```
            -9.8019802570343018e-01 1.</leafValues></_></weakClassifiers></_>
    <!-- stage 1 -->
    <_>
      <maxWeakCount>2</maxWeakCount>
      <stageThreshold>6.0084955766797066e-03</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 1 6.5239928662776947e-02</internalNodes>
          <leafValues>
            -9.5999997854232788e-01 1.</leafValues></_>
        <_>
          <internalNodes>
            0 -1 5 3.6031723022460938e-02</internalNodes>
          <leafValues>
            -9.8038423061370850e-01 9.6600848436355591e-
01</leafValues></_></weakClassifiers></_>
    <!-- stage 2 -->
    <_>
      <maxWeakCount>2</maxWeakCount>
      <stageThreshold>-9.6859291195869446e-02</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 4 3.8518323563039303e-03</internalNodes>
          <leafValues>
            8.4615385532379150e-01 -9.7894734144210815e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 2 9.7117483615875244e-02</internalNodes>
          <leafValues>
            -9.4301313161849976e-01 9.3385702371597290e-
01</leafValues></_></weakClassifiers></_>
    <!-- stage 3 -->
    <_>
      <maxWeakCount>1</maxWeakCount>
      <stageThreshold>9.5999997854232788e-01</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 0 1.8429078627377748e-03</internalNodes>
          <leafValues>
            9.5999997854232788e-01 -
1.</leafValues></_></weakClassifiers></_></stages>
  <features>
    <_>
      <rect>
        4 4 8 8 33</rect></_>
    <_>
      <rect>
        4 16 8 8 23</rect></_>
    <_>
      <rect>
        12 0 8 8 14</rect></_>
    <_>
      <rect>
        16 16 8 8 28</rect></_>
```

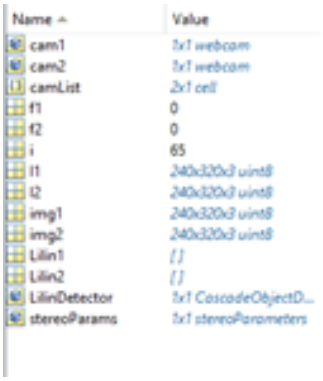
```
    <_>
      <rect>
        16 0 8 16 31</rect></_>
    <_>
      <rect>
        0 16 16 8 24</rect></_>
    <_>
      <rect>
        0 16 16 8 28</rect></_></features></cascade>
</opencv_storage>
```

Lampiran 2

2.1 Pengujian dengan objek bukan api



2.1.2 pengujian dengan lilin tidak berapi



2.2 Pengujian di Ruangan dengan intensitas 27 lux

2.2.1 Jarak 28 cm

Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[59,31,110,117]
Lilin2	[102,25,89,94]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[217.5000,165]
center2	[190.5000,79]
distanceAsString	0.28 meters'
distanceInMeters	0.2821
f1	472
f2	270
i	31
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[123.9501,584130,-2....
stereoParams	1x1 stereoParameters

2.2.2 Jarak 36 cm

Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[159,61,110,157]
Lilin2	[122,53,79,97]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[164.5000,183]
center2	[135,89.5000]
distanceAsString	0.36 meters'
distanceInMeters	0.3581
f1	412
f2	343
i	2000
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[123.9501,584130,-2....
stereoParams	1x1 stereoParameters

2.2.3 Jarak 46 cm

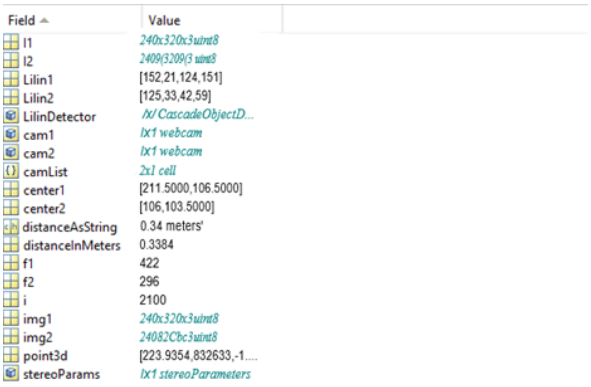
Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[159,61,110,157]
Lilin2	[122,53,79,97]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[193.5000,185]
center2	[154.5000,122]
distanceAsString	0.46 meters'
distanceInMeters	0.4572
f1	450
f2	370
i	1247
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[123.9501,584130,-2....
stereoParams	1x1 stereoParameters

2.2.4 Jarak 60 cm

Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[139,69,142,159]
Lilin2	[142,63,72,99]
LilinDetector	/x/ CascadeObjectD...
cam1	lx1 webcam
cam2	lx1 webcam
camList	2x1 cell
center1	[222.5000,100
center2	[179.5000,72.5000]
distanceAsString	0.59 meters'
distanceInMeters	0.5904
f1	427
f2	317
i	2315
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[323.9551,882530,-2....
stereoParams	lx1 stereoParameters

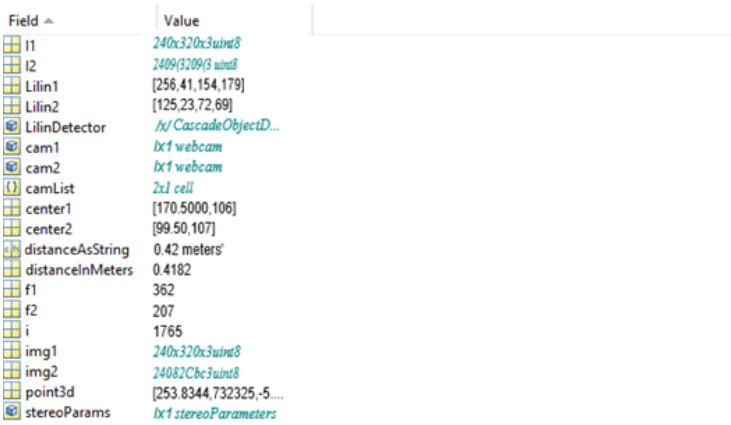
2.3 Pengujian di ruangan tanpa lampu dengan intensitas cahaya 7 lux

2.3.1 Jarak 30 cm



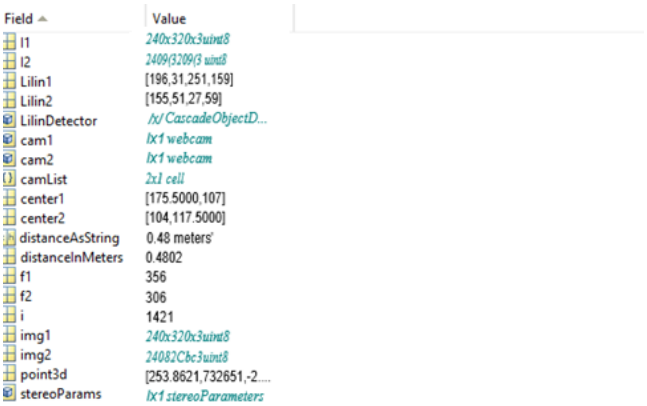
Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[152,21,124,151]
Lilin2	[125,33,42,59]
LilinDetector	/x/ CascadeObjectD...
cam1	lx1 webcam
cam2	lx1 webcam
camList	2x1 cell
center1	[211.5000,106.5000]
center2	[106,103.5000]
distanceAsString	0.34 meters'
distanceInMeters	0.3384
f1	422
f2	296
i	2100
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[223.9354,832633,-1....
stereoParams	lx1 stereoParameters

2.3.2



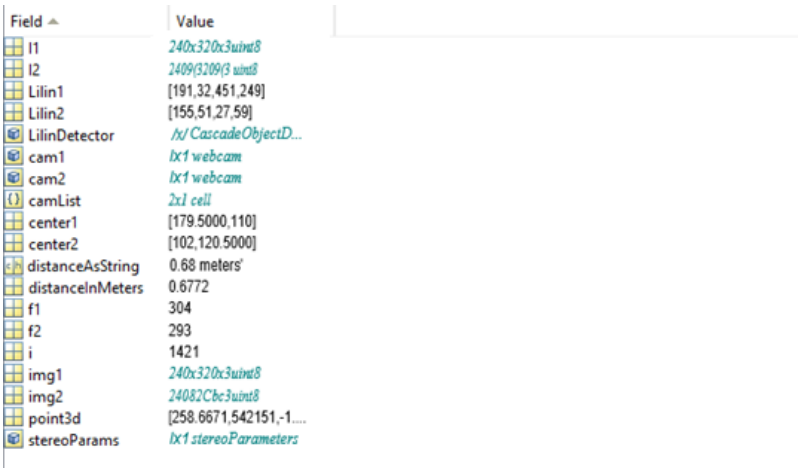
Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[256,41,154,179]
Lilin2	[125,23,72,69]
LilinDetector	/x/ CascadeObjectD...
cam1	lx1 webcam
cam2	lx1 webcam
camList	2x1 cell
center1	[170.5000,106]
center2	[99.50,107]
distanceAsString	0.42 meters'
distanceInMeters	0.4182
f1	362
f2	207
i	1765
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[253.8344,732325,-5....
stereoParams	lx1 stereoParameters

2.3.3



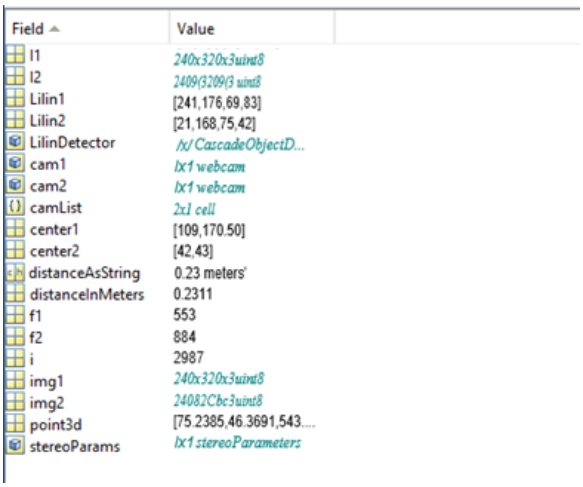
Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[196,31,251,159]
Lilin2	[155,51,27,59]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[175.5000,107]
center2	[104,117.5000]
distanceAsString	0.48 meters'
distanceInMeters	0.4802
f1	356
f2	306
i	1421
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[253.8621,732651,-2....
stereoParams	1x1 stereoParameters

2.3.4



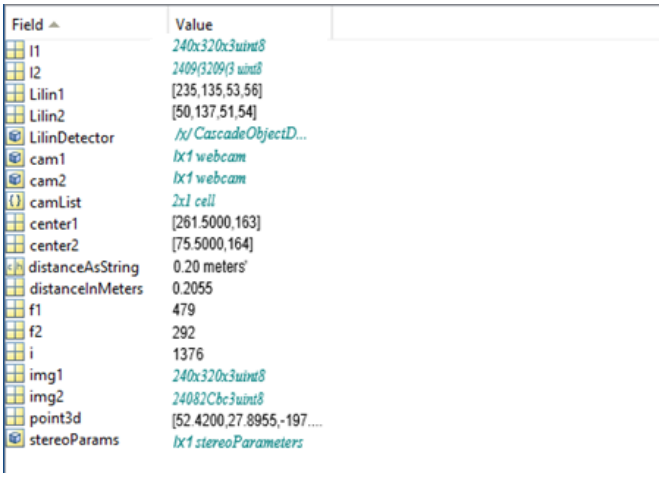
Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[191,32,451,249]
Lilin2	[155,51,27,59]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[179.5000,110]
center2	[102,120.5000]
distanceAsString	0.68 meters'
distanceInMeters	0.6772
f1	304
f2	293
i	1421
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[258.6671,542151,-1....
stereoParams	1x1 stereoParameters

2.4. Pengujian dengan dua buah api



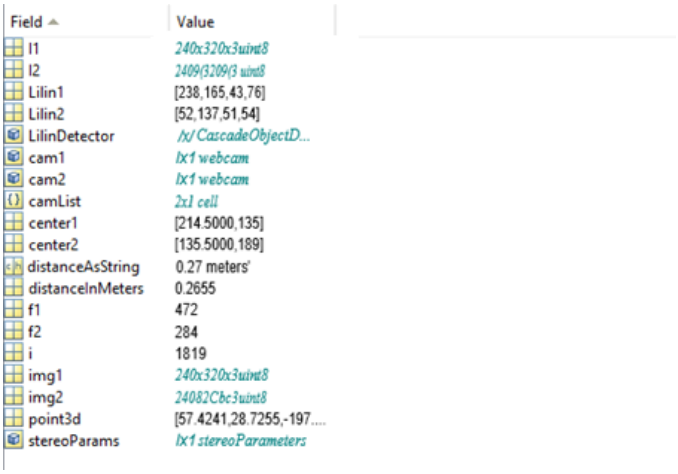
Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[241,176,69,83]
Lilin2	[21,168,75,42]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[109,170.50]
center2	[42,43]
distanceAsString	0.23 meters'
distanceInMeters	0.2311
f1	553
f2	884
i	2987
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[75.2385,46.3691,543....
stereoParams	1x1 stereoParameters

2.5.1 Jarak 20 cm



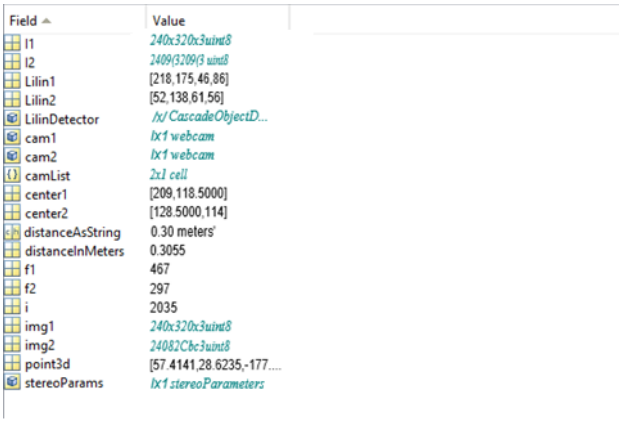
Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[235,135,53,56]
Lilin2	[50,137,51,54]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[261.5000,163]
center2	[75.5000,164]
distanceAsString	0.20 meters'
distanceInMeters	0.2055
f1	479
f2	292
i	1376
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[52.4200,27.8955,-197....
stereoParams	1x1 stereoParameters

2.5.2 Jarak 27 cm



Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[238,165,43,76]
Lilin2	[52,137,51,54]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[214.5000,135]
center2	[135.5000,189]
distanceAsString	0.27 meters'
distanceInMeters	0.2655
f1	472
f2	284
i	1819
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[57.4241,28.7255,-197....
stereoParams	1x1 stereoParameters

2.5.3 Jarak 30 xm



Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[218,175,46,86]
Lilin2	[52,138,61,56]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[209,118.5000]
center2	[128.5000,114]
distanceAsString	0.30 meters'
distanceInMeters	0.3055
f1	467
f2	297
i	2035
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[57.4141,28.6235,-177....
stereoParams	1x1 stereoParameters

2.5.4 Jarak 40 cm

Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[242,182,47,89]
Lilin2	[52,132,66,76]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[255,108.5000]
center2	[137,119]
distanceAsString	0.48 meters'
distanceInMeters	0.4765
f1	412
f2	277
i	2124
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[57.4241,28.6256,-137....
stereoParams	1x1 stereoParameters

2.6.1 Jarak 20 cm

Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[266,189,57,79]
Lilin2	[56,137,67,72]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[218.5000,172]
center2	[155.5000,177.5000]
distanceAsString	0.20 meters'
distanceInMeters	0.2024
f1	456
f2	310
i	2000
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[56.4893,28.6576,-137....
stereoParams	1x1 stereoParameters

2.6.2 Jarak 27 cm

Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[246,159,51,74]
Lilin2	[56,131,61,72]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[230,177.5000]
center2	[80.5000,177.5000]
distanceAsString	0.27 meters'
distanceInMeters	0.2674
f1	423
f2	316
i	2244
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[56.7893,28.6846,-157...
stereoParams	1x1 stereoParameters

2.6.3 Jarak 30 cm

Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[227,153,55,72]
Lilin2	[57,134,65,78]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[258,182.5000]
center2	[155.5000,172.5000]
distanceAsString	0.32 meters'
distanceInMeters	0.3162
f1	467
f2	361
i	2000
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[52.7853,28.6876,-137....
stereoParams	1x1 stereoParameters

2.6.4 Jarak 40 cm

Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[236,153,55,72]
Lilin2	[59,134,65,78]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[168,272]
center2	[85.5000,179]
distanceAsString	0.47 meters'
distanceInMeters	0.4660
f1	410
f2	344
i	2976
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[52.6153,28.6876,-177....
stereoParams	1x1 stereoParameters

2.7 Pengujian di Ruangan 3

2.7.1 JARAK 20CM

Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[282,153,59,72]
Lilin2	[54,137,67,78]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[260,182.5000]
center2	[58.5000,172]
distanceAsString	0.22 meters'
distanceInMeters	0.2172
f1	419
f2	387
i	1455
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[52.6153,28.6876,-177...
stereoParams	1x1 stereoParameters

2.7.2 Jarak 30 cm

Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[212,113,61,72]
Lilin2	[52,137,57,68]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[205,108.5000]
center2	[127.5000,116]
distanceAsString	0.34 meters'
distanceInMeters	0.3372
f1	388
f2	315
i	1721
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[55.6653,27.6896,-127....
stereoParams	1x1 stereoParameters

2.7.3 Jarak 40 cm

Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[255,112,61,72]
Lilin2	[58,137,57,68]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[218.5000,103]
center2	[131.5000,121.5000]
distanceAsString	0.39 meters'
distanceInMeters	0.3922
f1	372
f2	312
i	1899
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[55.6753,28.7296,-127....
stereoParams	1x1 stereoParameters

2.7.4 Jarak 50 cm

Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[154,122,67,74]
Lilin2	[52,138,59,88]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[253.5000,118]
center2	[125.5000,115]
distanceAsString	0.52 meters'
distanceInMeters	0.5168
f1	402
f2	355
i	2454
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[65.6743,27.7226,-137....
stereoParams	1x1 stereoParameters

2.5 Pengujian di Ruangan 4

2.8.1 Jarak 20 cm

Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[206,138,64,68]
Lilin2	[29,149,53,57]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[238,172]
center2	[55.5000,177.5000]
distanceAsString	0.20 meters'
distanceInMeters	0.2044
f1	476
f2	288
i	2000
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[43.3165,33.4548,203.....
stereoParams	1x1 stereoParameters

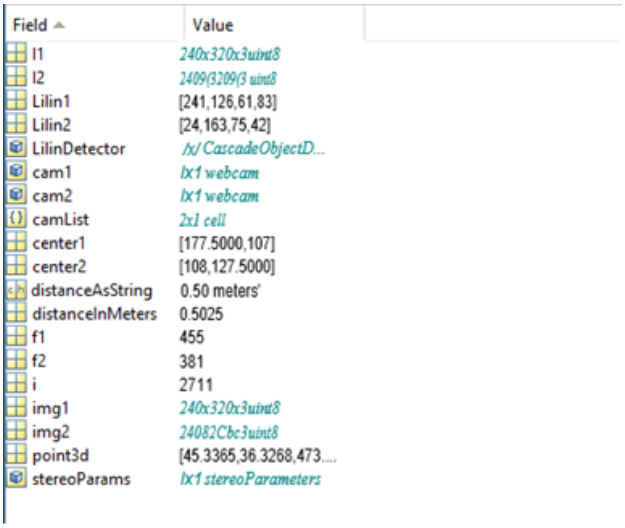
2.8.2 Jarak 30 cm

Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[227,161,67,62]
Lilin2	[29,112,56,59]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[281,122.5000]
center2	[142.5000,116]
distanceAsString	0.30 meters'
distanceInMeters	0.3021
f1	412
f2	246
i	2022
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[48.3365,36.4248,443....
stereoParams	1x1 stereoParameters

2.8.3 Jarak 40 cm

Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[261,171,62,81]
Lilin2	[29,182,76,49]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[175.5000,106]
center2	[99.5000,117]
distanceAsString	0.41 meters'
distanceInMeters	0.4081
f1	445
f2	376
i	2782
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[45.3365,36.3268,473....
stereoParams	1x1 stereoParameters

2.8.4 Jarak 50 cm



Field	Value
I1	240x320x3uint8
I2	2409(3209(3 uint8
Lilin1	[241,126,61,83]
Lilin2	[24,163,75,42]
LilinDetector	/x/ CascadeObjectD...
cam1	1x1 webcam
cam2	1x1 webcam
camList	2x1 cell
center1	[177.5000,107]
center2	[108,127.5000]
distanceAsString	0.50 meters'
distanceInMeters	0.5025
f1	455
f2	381
i	2711
img1	240x320x3uint8
img2	24082Cbc3uint8
point3d	[45.3365,36.3268,473....
stereoParams	1x1 stereoParameters