RANCANG BANGUN SISTEM PENDETEKSIAN DIMENSI OBYEK MENGGUNAKAN METODE HARRIS CORNER DAN LUCAS KANADE BERBASIS CITRA STEREO

(Skripsi)

Oleh

WINAL PRAWIRA

FAKULTAS TEKNIK

UNIVERSITAS LAMPUNG

2017

ABSTRACT

THE DETECTION OF 3D OBJECT USING A METHOD OF A HARRIS CORNER DETECTOR AND LUCAS-KANADE TRACKER BASED ON STEREO IMAGE

By

WINAL PRAWIRA

This research proposes the use of Harris Corner Detector and Lucas-Kanade Tracker methods for the detection of 3D objects based on stereo image. This image is a result of camera capture from various objects such as tubes, balls, cubes, and 2-dimensional images. This research is the early step in the development of the ability of a computer vision to be able to mimic the performance of eye organs in humans for detecting an object.

This research will detect three-dimensional objects, while for two-dimensional objects will be ignored and considered as part of the background. The method used in this research is Harris Corner and Lucas Kanade. The detection step in this system starts by determining feature point on image result from stereo camera use Harris Corner detector, and after the feature point of the two images obtained, then performed corresponding using Lucas Kanade whose results will be combined and determine the dimensions of the object being observed.

The Effectiveness of the detection result of the proposed method is measured using the recall and precision parameter values obtained in the merged of the image. And in this research the average value of recall and precision on ball , cube, and tube objects above 50% from image result with distance between the cameras 10, and 20cm. While on the distance between the cameras 50cm in all objects produced the value of recall and precision below 50%.

Keywords: Harris Corner, Lucas Kanade, Object Dimension, Image Detection, Stereo Image.

ABSTRAK

RANCANG BANGUN SISTEM PENDETEKSIAN DIMENSI OBYEK MENGGUNAKAN METODE HARRIS CORNER DAN LUCAS KANADE BERBASIS CITRA STEREO

Oleh

WINAL PRAWIRA

Penelitian yang dilakukan tentang pendeteksian dimensi obyek menggunakan metode Harris Corner dan Lucas Kanade berbasis citra stereo ini didapat dari hasil tangkapan kamera dengan berbagai obyek seperti tabung, bola, kubus, serta gambar 2 dimensi. Penelitian ini merupakan langkah awal dalam pengembangan kemampuan *computer vision* untuk meniru kinerja organ mata pada manusia dalam mendeteksi sebuah obyek secara otomatis.

Sistem akan mendeteksi obyek berdimensi tiga, sedangkan untuk obyek yang berdimensi dua akan diabaikan dan dianggap sebagai bagian dari latar belakang.

Langkah pendeteksian yang terdapat dalam sistem ini dimulai dengan menentukan titik point pada citra hasil stereo kamera dan dikorespondensikan menggunakan Lucas Kanade yang hasilnya nanti akan digabungkan dan menentukan dimensi obyek benda yang diamati. Hasil pendeteksian diukur tingkat ketepatan dan ketelitiannya menggunakan nilai *recall* dan *precision*.

Dan pada penelitian ini hasil rata rata nilai *recall* dan *precision* pada obyek bola, kubus, dan tabung diatas 50% pada jarak antar kamera 10, dan 20cm, sedangkan pada jarak antar kamera 50cm disemua objek dihasilkan nilai *recall* dan *precision* dibawah 50%.

Kata Kunci : Harris Corner, Lucas Kanade, Dimensi Obyek, Pendeteksian Citra, Citra Stereo.

RANCANG BANGUN SISTEM PENDETEKSIAN DIMENSI OBYEK MENGGUNAKAN METODE HARRIS CORNER DAN LUCAS KANADE BERBASIS CITRA STEREO

Oleh

WINAL PRAWIRA

Skripsi
Sebagai Salah Satu Syarat Untuk Mencapai Gelar
SARJANA TEKNIK

Pada

Jurusan Teknik Elektro
Fakultas Teknik Universitas Lampung

FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2017

Judul Skripsi	:	**RANCANG BANGUN SISTEM PENDETEKSIAN DIMENSI OBYEK MENGGUNAKAN METODE HARRIS CORNER DAN LUCAS KANADE BERBASIS CITRA STEREO**
Nama Mahasiswa	:	WINAL PRAWIRA
Nomor Pokok Mahasiswa	:	1215031076
Program Studi	:	Teknik Elektro
Fakultas	:	Teknik

MENYETUJUI

1. Komisi Pembimbing

Ir. Emir Nasrullah, S.T.,M.Eng	**Dr. Eng. F. X. Arinto S, S.T.,M.T.**
NIP 196006141994021001	NIP : 196912191999031002

2. Ketua Jurusan Teknik Elektro

Dr.Ing. Ardian Ulvan, S.T., M.Sc.
NIP : 19731128 199903 1 005

MENGESAHKAN

1. Tim Penguji

Ketua : **Ir. Emir Nasrullah, S.T.,M.Eng** .....

Sekretaris : **Dr. Eng. F. X. Arinto S, S.T.,M.T.** .....

Penguji :
BukanPembimbing **Dr. Ir. Sri Ratna S, M.T.** .....

2. Dekan Fakultas Teknik

Prof. Suharno, M.Sc., Ph.D
NIP 19620717 198703 1 002

Tanggal Lulus Ujian Skripsi : **16 Juni 2017**

SURAT PERNYATAAN KEASLIAN HASIL KARYA

Saya yang bertanda tangan dibawah ini, menyatakan bahwa skripsi saya yang berjudul "RANCANG BANGUN SISTEM PENDETEKSIAN DIMENSI OBYEK MENGGUNAKAN METODE HARRIS CORNER DAN LUCAS KANADE BERBASIS CITRA STEREO" merupakan hasil karya sendiri dan bukan hasil karya orang lain. Semua hasil yang tertuang dalam skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Apabila pernyataan saya tidak benar, dan dikemudian hari terbukti bahwa skripsi ini merupakan Salinan atau dibuat oleh orang lain maka saya bersedia dikenai sangsi sesuai dengan ketentuan akademik yang berlaku.

Bandar Lampung, Juli 2017

(Winal Prawira)
PM 1215031076

RIWAYAT HIDUP

Penulis dilahirkan di Bandar Jaya, Lampung Tengah pada tanggal 29 November 1993 sebagai anak pertama dari tiga bersaudara, dari bapak Zainal Abidin dan ibu Wiyatun. Pendidikan sekolah dasar diselesaikan di SDN 1 Karang Maritim pada tahun 2006, Sekolah Menengah Pertama di SMPN 29 Bandar Lampung diselesaikan pada tahun 2009, dan Sekolah Menengah Kejuruan di SMKN 2 Bandar Lampung diselesaikan pada tahun 2012. Pada tahun 2012, penulis terdaftar sebagai mahasiswa Jurusan Teknik Elektro Fakultas Teknik Universitas Lampung melalui jalur SNMPTN (Seleksi Nasional Masuk Perguruan Tinggi Negeri) 2012. Selama menjadi mahasiswa penulis aktif menjadi asisten pada tahun 2014 hingga 2016 di Laboratorium Elektronika. Penulis juga aktif dalam organisasi Himpunan Mahasiswa Teknik Elektro (HIMATRO) sebagai Kepala Divisi Pengabdian Masyarakat pada tahun 2014-2015. Pada rentang waktu 01 September-30 September 2015, penulis melaksanakan Kerja Praktik di PT.Dirgantara Indonesia dan ditempatkan pada Divisi Pusat Teknologi, Departemen Analisa Sistem, Bidang *Avionics & Flight Deck System*. Penyelesaian Kerja Praktik tersebut menghasilkan sebuah laporan Kerja Praktik dengan judul "Prinsip Kerja *Traffic Alert & Collision Avoidance System* Sebagai Bagian Dari *Situation Awareness* Pada Pesawat N219".

PERSEMBAHAN

Dengan Ridho Allah SWT, teriring shalawat kepada Nabi Muhammad SAW

Karya tulis ini kupersembahkan untuk:

Ayah dan Ibuku Tercinta
Zainal Abidin & Wiyatun

Serta Adik adikku Tersayang
Galih Sandria dan Aztri Alvi Syahrin

Teman-teman kebanggaanku
Rekan-rekan Jurusan Teknik Elektro

Almamaterku
Universitas Lampung

Bangsa dan Negaraku
Republik Indonesia

Terima-kasih untuk semua yang telah diberikan kepadaku. *Jazzakallah Khairan.*

MOTTO

"Karena Sesungguhnya sesudah kesulitan itu ada kemudahan.Sesungguhnya sesudah kesulitan itu ada kemudahan."
(Al-Quran, Surat Al – Insyirah, 94 : 5 – 6)

"Tidak ada balasan untuk kebaikan selain kebaikan (pula)"
(Al-Quran, Surat Ar – Rahman, 55 : 60)

"Tak perlu bersikeras menjelaskan siapa dirimu, karena orang yang mencintaimu tak membutuhkan itu, dan orang yang membencimu tak akan percaya itu"
(Ali bin Abi Thalib RA)

"Jika kamu mengetahui tak banyak masalah yang dapat kamu atasi secara ahli, yang kamu butuh hanya percobaan yang lebih banyak dari orang yang ahli, dan jangan menyerah saat menghadapai kegagalan"
(Anonymus)

SANWACANA

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat dan hidayahnya kepada penulis, sehingga dapat menyelesaikan tugas akhir ini. Shalawat serta salam disanjungkan kepada Nabi Muhammad Shalallahu Alaihi Wassalam yang dinantikan syafaatnya di hari akhir kelak.

Tugas akhir ini berjudul "**RANCANG BANGUN SISTEM PENDETEKSIAN DIMENSI OBYEK MENGGUNAKAN METODE HARRIS CORNER DAN LUCAS KANADE BERBASIS CITRA STEREO**" digunakan sebagai salah satu syarat guna memperoleh gelar sarjana di jurusan Teknik Elektro Fakultas Teknik Universitas Lampung.

Dalam masa perkuliahan dan penelitian, penulis mendapat banyak hal baik berupa dukungan, semangat, motivasi dan banyak hal yang lainya. Untuk itu penulis mengucapkan terimakasih kepada :

1. Bapak Prof. Dr. Ir. Hasriadi Mat Akin, M.P. selaku Rektor Universitas Lampung.
2. Bapak Prof. Suharno, M.Sc., Ph.D. selaku Dekan Fakultas Teknik Universitas Lampung.
3. Bapak Dr. Ing. Ardian Ulvan, S.T., M.Sc. Selaku kepala Jurusan Teknik Elektro fakultas Teknik Universitas Lampung.

4. Bapak Ir. Emir Nasrullah, M.Eng selaku dosen pembimbing utama terimakasih atas kesedian waktunya untuk membimbing dan memberikan ilmu.
5. Bapak Dr. Eng. F.X. Arinto S, S.T.,M.T. selaku pembimbing kedua terimakasih atas waktu, pengalaman, ilmu, dan bimbingannya selama mengerjakan tugas akhir.
6. Ibu Dr. Ir. Sri Ratna S, M.T. selaku dosen penguji tugas akhir, penulis sangat berterima kasih atas pengalaman hidup dan masukan yang telah diberikan dalam masa perkuliahan dan penelitian guna membuat tugas akhir ini menjadi lebih baik.
7. Seluruh Dosen Teknik Elektro, Terimakasih atas bimbingan dan ilmu yang telah diberikan selama menuntut ilmu di Jurusan Teknik Elektro Universitas Lampung.
8. Keluarga Besar Teknik Elektro, Mbak Ning, Mbak Dea, Mas Daryono , dll, terimakasih atas kebersamaan dan waktu serta ilmu yang telah diberikan.
9. Kepada seseorang yang telah memberikan semangat agar penulis segera menyelesaikan tugas akhir ini. Terima kasih atas canda tawa, dan segala pelajaran hidup yang telah diberikan.
10. Para sahabat yang telah menemani dan memotivasi penulis agar selalu memberikan yang terbaik dalam segala hal, Dema, Esha, Wahyu, Dedy, Marezkha, Mia.
11. Keluarga KKN Desa Waspada, Daru, Chan, Jalu, Kiki, Ike, Hanny, Yumi, Tiara, dan Gita terimakasih atas pengalaman, semangat, motivasi, serta hal-hal yang telah membuat penulis semangat untuk mengerjakan Tugas Akhir ini.

12. Terimakasih kepada teman seperjuangan atas semua kenangan indah yang menemani penulis menyelesaikan Tugas Akhir ini, Yogi Aldino, Kris Sivam, Faizun Iqbal Zulfi, doaku untuk kebaikan kalian, dan maaf banyak merepotkan selama ini.
13. Seluruh punggawa, dan asisten Laboratorium Teknik Elektronika : Kak Jerry, Kak Yudi, Kak Andri, Kak Victor, Kak Agung, Mba Eliza, Kak Frisky, Kak Abidin, Mba Layla, Kak Subas, Bella, Windy, Desi, Gusti, Nando, Reza, Roy, Inyong, Jul, Ketut, Rafi, Tama, Ridho, Rico, Pami, Alam, Mbah, Al, Arif F, Dapin otong, Dinda, Andre P, Aggi, tiya, osline, cintia, Terima kasih atas canda tawa dan kebersamaannya di dalam keluarga Laboratorium Teknik Elektronika.
14. Seluruh Asisten Laboratorium Terpadu Teknik Elektro Universitas Lampung, khususnya, pinggiran roti, billy, mail, eko, egi, ade, gita, nurul, niken, yona, widia, dan yang lainnya terima kasih atas dukungan dan bantuannya selama ini.
15. Teman-teman keluarga besar ELANG (Elektro Angkatan) 2012 Mbah, fiki, salam, ipan, albana, anju, bembeng, ucok, Amanda, dika, fahrur, fikri, ghum, ghifin, halim, soni, robani, maqbull, adnan, yono, nyomen, ratih, risda, taufiq, windu, rio, aji, dharma, guntur, yayan, panji, agung, aji I, alandani, andri, angga, kocong, tiyar, didi, eko, erwin, reza, isol, irul, mahe, rama, vincent, wanto, komti, rahmat terimakasih atas segala yang telah diberikan, kehangatan keluarga ini memang luar biasa.

Semoga apa yang telah diberikan selama ini mendapat balasan yang lebih baik dari dari Allah SWT. Penulis meminta maaf atas segala kesalahan dan ketidak sempurnaan dalam penulisan Tugas Akhir ini. Kritik dan saran yang membangun sangat diharapkan agar dapat dimanfaatkan dan dikembangkan dimasa mendatang.

Bandar lampung, Juli 2017

Penulis

Winal Prawira

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Gambar ..... Halaman

DAFTAR TABEL

I. PENDAHULUAN

1.1 Latar Belakang

Kemajuan ilmu teknologi pengolahan citra digital (*Digital Image Processing*) yang semakin pesat diharapkan dapat mempermudah kehidupan manusia dalam berbagai bidang, mulai dari bidang militer, bidang kedokteran hingga pada bidang rumah tangga. Teknologi otomasi menerapkannya untuk menghilangkan peran manusia dalam pengambilan keputusan.

Pengolahan citra adalah teknik mengolah citra yang mentransformasikan citra masukan agar keluarannya memiliki kualitas yang lebih baik dibandingkan kualitas citra masukan, dan bermanfaat dalam kehidupan kita seperti untuk meningkatkan kualitas citra, menghilangkan cacat pada citra, mengidentifikasi objek, dan penggabungan dengan bagian citra yang lain.

Penelitian ini memanfaatkan teknologi tersebut, untuk menangkap suatu obyek yang ada di depan kamera dan mampu mengidentifikasi jenis dimensi dari objek tersebut.

Penelitian ini menggunakan software pendukung *openCV* supaya memudahkan penggguna dalam identifikasi objek secara sederhana. Identifikasi pada obyek 3 dimensi dilakukan dari dua buah gambar 2 dimensi yang telah diekstrak dari format *avi* ke format *bmp*, proses pengambilan citra menggunanakan dua buah kamera ini lebih dikenal dengan istilah

stereovision. Kemudian gambar 2 dimensi hasil dari kamera kiri dan kanan diolah menggunakan metode *Harris corner* terlebih dahulu guna mendapatkan titik titik yang kita inginkan.

Selanjutnya setelah mendapatkan citra dengan masing masing titiknya, citra diolah kembali menggunakan metode *Lucas kanade*, metode ini merupakan suatu metode yang menghubungkan kesamaan atau korelasi antara titik titik poin pada satu citra pada citra lainnya.

Hasil dari sistem inilah yang nantinya akan dijadikan acuan dan kemudian divisualisasikan ke dalam bentuk *template* atau *bounding box* jenis dimensi objek yang sesuai dengan target objek yakni 2 dimensi atau 3 dimensi.

Hasil dari tugas akhir ini bertujuan untuk membantu masyarakat dalam mendefinisikan objek disekitarnya menggunakan teknologi yang lebih modern, praktis, dan otomatis.

Penelitian ini mengacu pada pengembangan hasil dari penelitian sebelumnya seperti pada tabel 1.1 yang dilakukan oleh (Rachmawati, Risanuri Hidayat, Sunu Wibirama.) yang telah membuat aplikasi komputer yang dapat merekonstruksi obyek tiga dimensi dari citra dua dimensi menggunakan *epipolar geometry*.

Tabel 1.1 Penelitian Sebelumnya

No	Nama (Instansi)	Tahun	Judul penelitian
1	Sumantra Dutta Roy (Indian Institute of Technology Bombay, Powai, Mumbai)	2001	*Active 3-D Object Recognition through Next View Planning*
2	M. Y. Mashor, M. K. Osman, M. R. Arshad (Electronic & Biomedical Intelligent Sistems (EBItS) Research Group, Control and ELectronic Intelligent Sistems (CELIS) Research Group.)	2006	*3D Object Recognition Using Multiple Views and Neural Networks*
3	Rachmawati, Risanuri Hidayat, Sunu Wibirama (Jurusan Teknik Elektro, Fakultas Teknik Universitas Gajah Mada)	2012	Rekonstruksi Obyek Tiga Dimensi Dari Citra Dua Dimensi Menggunakan *Epipolar Geometry*

1.2 Tujuan Penelitian

Tujuan dari penelitian ini antara lain :

1. Membuat sistem pendeteksian pola dua dimensi serta tiga dimensi menggunakan metode *Harris corner, Lucas kanade,* dan *Stereovision.*
2. Mengetahui apakah metode yang digunakan dapat dioperasikan menggunakan software *OpenCV* dan *Visual Studio*.

1.3 Manfaat Penelitian

Manfaat dari penelitian ini yaitu :

Didapatkan sebuah sistem pengolahan citra yang mampu mendeteksi dimensi dari obyek yang diamati, sehingga nantinya sistem ini dapat dikembangkan menjadi sebuah *software* pengawasan seperti *CCTV* yang dapat digunakan secara *portable* dan fleksibel, serta mampu melakukan proses identifikasi secara otomatis.

1.4 Rumusan Masalah

Rumusan masalah dari penelitian ini adalah :

- Bagaimana merancang sistem identifikasi dimensi obyek menggunakan metode *stereovision* ?
- Bagaimana mengolah citra gambar masukan agar mampu dijadikan informasi guna mengidentifikasi dimensi obyek.
- Bagaimana akurasi identifikasi sistem dengan penggabungan dua buah citra masukan menggunakan metode yang digunakan
- Bagaimana hasil akurasi identifikasi untuk dimensi obyek 2 dimensi dan 3 dimensi

1.5 Batasan Masalah

Pada penelitian ini dibatasi pada hal-hal berikut :

1. Bentuk obyek yang diamati meliputi variasi bentuk 3 dimensi sederhana seperti tabung, bola, dan kubus. Sedangkan untuk obek 2 dimensi adalah beberapa gambar 2 dimensi.
2. Software yang digunakan untuk pengolahan citra adalah software *OpenCV* dan *Visual Studio*.
3. Kamera yang digunakan untuk pengambilan data dalah jenis *Car Dashboard Camera*.
4. Citra yang menjadi masukan merupakan 2 citra yang memiliki objek sama tetapi memiliki sudut pandang yang berbeda dalam rentang waktu pengambilan yang sama. Referensi yang dipakai adalah citra kiri
5. Jarak antar kamera yang digunakan yaitu : 10cm, 20cm, dan 50cm.

1.6 Hipotesis

Sistem yang dirancang diduga dapat melakukan pendeteksian terhadap jenis dimensi obyek yang diamati. Penggunaan software *OpenCV* dan *Visual Studio* juga diduga dapat digunakan dalam proses pendeteksian tersebut. Hasil pengolahan data diduga dapat mengelompokan dimensi suatu obyek secara otomatis.

1.7 Sistematika Penulisan

Untuk memudahkan penulisan dan pemahaman mengenai materi tugas akhir ini, maka tulisan akan dibagi menjadi lima bab, yaitu:

BAB I. Pendahuluan

Memuat latar belakang masalah, tujuan, rumusan masalah, batasan masalah, manfaat penelitian, hipotesis, dan sistematika penulisan.

BAB II. Tinjauan Pustaka

Memuat landasan teori yang digunakan dalam penelitian dan membahas penelitian yang telah dan akan dilakukan berhubungan dengan penelitian.

BAB III. Metode Penelitian

Memuat langkah-langkah penelitian yang dilakukan diantaranya waktu dan tempat penelitian, alat dan bahan, dan tahap-tahap perancangan.

BAB IV. Hasil dan Pembahasan

Bab ini berisi hasil pengujian dan pembahasan terhadap kinerja alat atau sistem yang telah dirancang.

BAB V. Kesimpulan dan Saran

Memuat kesimpulan dan saran tentang penelitian yang telah dilakukan.

II. TINJAUAN PUSTAKA

2.1 Citra

Citra merupakan fungsi kontinu dari intensitas cahaya pada suatu bidang. Sebagian dari berkas cahaya yang berasal dari sumber cahaya akan dipantulkan kembali, pantulan cahaya inilah yang ditangkap oleh alat optik, misalnya mata pada manusia, kamera, pemindai, dan sebagainya, sehingga bayangan objek yang disebut citra tersebut terekam (Hermawan, 2014). Citra yang biasa dibicarakan merupakan contoh dari "citra diam" yaitu citra tunggal yang tidak bergerak. Sedangkan citra yang bergerak adalah rangkaian citra diam yang ditampilkan secara beruntun (sekuensial) sehingga memberi kesan pada mata sebagai gambar yang bergerak dan disebut *frame*.

2.1.1 Definisi Pengolahan Citra

Sebuah citra sudah pasti kaya informasi, namun seringkali citra yang diperoleh mengalami penurunan mutu. Citra semacam ini menjadi lebih sulit diinterpretasi karena informasi yang disampaikan oleh citra tersebut menjadi berkurang. Agar citra mudah diinterpretasi maka citra tersebut perlu dimanipulasi menjadi citra yang kualitasnya lebih baik. Terminologi lain yang berkaitan erat dengan pengolahan citra adalah *computer vision*. Pada hakikatnya, *computer vision* mencoba meniru

cara kerja sistem visual manusia (Hadwi, 2013). Manusia sendiri melihat objek dengan indera penglihatan, lalu citra objek diteruskan ke otak untuk diinterpretasi sehingga manusia mengerti objek apa yang tampak dalam pandangan matanya.

Hasil interpretasi ini digunakan untuk pengambilan keputusan, misalnya digunakan untuk memberi perintah menghindar ketika melihat halangan saat sedang berjalan. *Computer vision* merupakan proses otomatis yang mengintegrasikan sejumlah besar proses untuk persepsi visual, seperti akuisisi citra, pengolahan citra, klasifikasi, pengenalan (*recognition*), dan membuat keputusan (Mulyawan, 2014).

Operasi-operasi yang dilakukan di dalam pengolahan citra banyak. ragamnya. Namun, secara umum, operasi pengolahan citra dapat diklasifikasikan dalam beberapa jenis sebagai berikut :

a) Perbaikan Kualitas Citra (*Image Enhancement*)
b) Pemugaran citra (*image restoration*).
c) Pemampatan citra (*image compression*).
d) Segmentasi citra (*image segmentation*).
e) Analisa citra (*image analysis*)
f) Rekonstruksi citra (*image reconstruction*)

2.2 Format File Citra

Citra digital memiliki beberapa jenis format file yang dapat digunakan serta memiliki karakteristik dan manfaat yang berbeda dari tiap format file citra digital yang digunakan. Format file citra digital dapat berupa .bmp, *.bmp*, .png, .tiff, .gif, .pgm, dan lain-lain. Sedangkan untuk format video ada beberapa macam jenis antara lain ASF, *AVI*, DV, mpg, vob, mov, mp4, wmf, dan 3gp. Dalam tugas akhir ini digunakan format file citra digital *bmp*, serta format file video *avi*.

2.2.1 Citra Bitmap (bmp)

Bitmap merupakan representasi citra grafis yang terdiri dari susunan titik titik yang tersimpan di memori komputer. Dikembangkan oleh Microsoft, nilai dari setiap titik pada format bmp diawali oleh satu bit data untuk gambar hitam putih, atau lebih dari satu bit saat gambar memiliki warna lainnya. Ukuran sebenarnya untuk **n**-bit (2^n warna) bitmap dalam byte dapat dihitung :

$$\textbf{ukuran file BMP} \approx 54 + 4 \cdot 2^n + \frac{lebar \cdot tinggi \cdot n}{8},$$

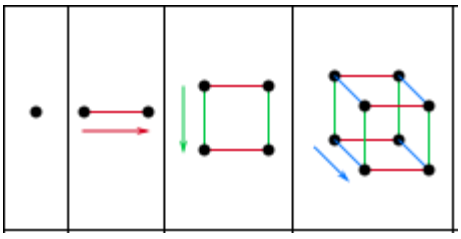
Kerapatan titik-titik tersebut dinamakan resolusi, yang menunjukkan seberapa tajam gambar ini ditampilkan, ditunjukkan dengan jumlah baris dan kolom, contohnya 1024x768. Beberapa format file bitmap yang populer adalah BMP, PCX dan TIFF.

2.2.2 *AVI* (*Audio Video Interleaved*)

Audio Video Interleaved adalah format standar file video untuk Microsoft Windows. Format video ini mampu menghasilkan pergerakan 15 *fps* dengan kualitas suara mencapai 11,025Hz. Hampir semua kamera video, khususnya yang analog, menghasilkan format file berekstensi "*avi"* saat ditransfer ke PC.

2.3 Dimensi Obyek

Secara umum pengertian dimensi adalah parameter atau pengukuran yang dibutuhkan untuk mendefinisikan sifat-sifat suatu objek yaitu panjang, lebar dan tinggi atau ukuran dan bentuk. Dalam matematika dan fisika, dimensi adalah parameter yang dibutuhkan untuk menggambarkan posisi dan sifat-sifat objek dalam suatu ruang. Dua dimensi adalah bentuk dari benda yang memiliki panjang dan lebar, sedangkan 3 dimensi adalah bentuk dari benda yang memiliki panjang, lebar, dan tinggi. Benda-benda dimensi tiga sering juga disebut dengan istilah bangun ruang (Pujiyanta, 2009). Pada penelitian ini penulis membatasi dimensi obyek yang akan diamati antara lain benda dengan dimensi dua dan tiga saja.

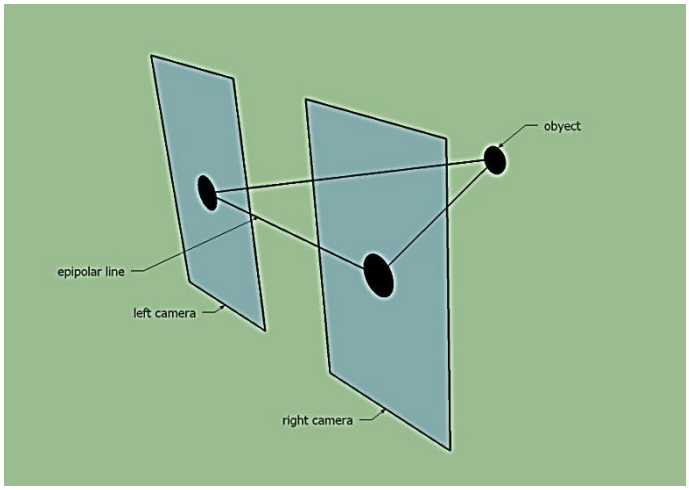


Gambar 2.1. Contoh Dimensi Benda

2.4 Streovision

Stereovision merupakan suatu usaha untuk mendapatkan citra stereo dari suatu objek dari dua posisi yang berbeda. Citra stereo didapatkan dengan cara meletakkan dua kamera dibidang yang sama dengan jarak tertentu. Jarak antar kamera bergantung dari jarak objek terdekat yang diamati dan juga dari tingkat citra stereo yang diinginkan (Hadwi, 2013). Terdapat dua metode dalam peletakan kamera :

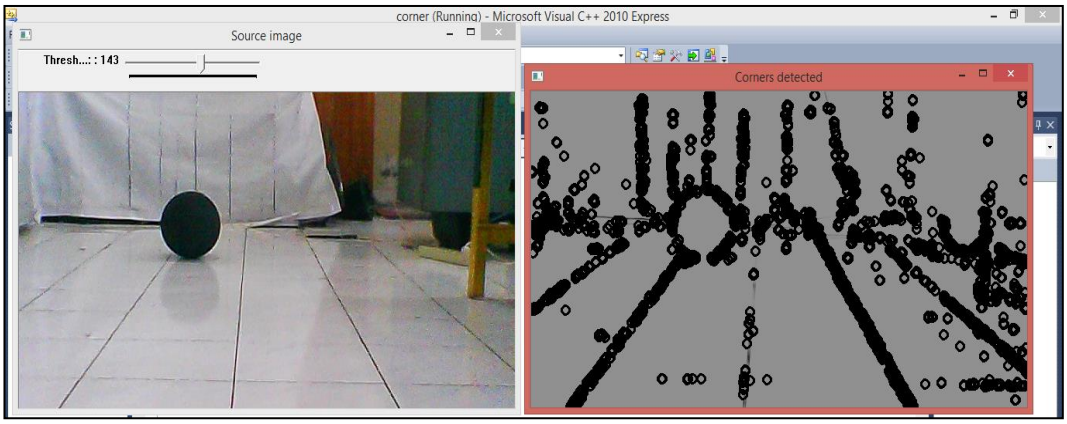
- Peletakan kamera dimana fokus dari kedua kamera berada pada titik api (focal) yang sama terhadap objek yang diamati. Dalam metode ini lebar dari daerah yang diamati pada titik apinya (focal) tidak sama.
- Peletakan kamera dengan lebar dari daerah yang diamati sama, dimana fokus dari masing-masing kamera tidak berada pada pusat focal yang sama tetapi masih dalam satu garis focal yang sama. Sehingga lebar daerah yang diamati sama besar pada titik apinya (focal).



Gambar 2.2. Stereovision

2.5 Harris Corner Detection

Harris Corner detector adalah detektor titik (sudut) yang populer karena mampu menghasilkan nilai yang konsisten walau dengan adanya rotasi, skala, variasi pencahayaan maupun noise pada gambar (Sany, 2015). Detektor sudut Harris didasarkan pada fungsi autokorelasi sinyal lokal di mana fungsi autokorelasi lokal akan menghitung perubahan lokal dari sinyal. Titik sudut tidak akan bisa didefinisikan pada piksel tunggal, karena disana hanya ada satu gradien per titik (Solihin,2013). Perilaku gradien ini jika kita cuplik dalam sebuah jendela kecil dapat dikategorikan bedasarkan statistiknya sebagai berikut : Konstan: Jika hanya sedikit atau tidak ada perubahan kecerahan, sisi/tepi/garis : Jika terjadi perubahan kecerahan yang kuat pada satu arah, Flow : garis Paralel, Pojok (corner) : Jika terjadi perubahan kecerahan yang kuat dalam arah orthogonal



Gambar 2.3. Contoh Metode *Harris corner* Detection

2.6 Lucas kanade

Dalam *computer vision*, metode *Lucas kanade* adalah metode differensial yang banyak digunakan untuk melakukan estimasi optical flow.

Dikembangkan oleh Bruce D. Lucas dan Takeo Kanade. Metode ini mengasumsikan bahwa aliran/flow pada dasarnya konstan di lingkungan lokal dari piksel yang dipertimbangkan dan memecahkan persamaan aliran dasar optik di semua lingkungan tersebut dengan kriteria kuadrat terkecil (Arinto, 2014). Metode *Lucas kanade* mengasumsikan bahwa perpindahan objek dari dua *frame* yang kecil dan konstan dalam lingkungan titik p yang menjadi pertimbangan. Jadi persamaan optik dapat dianggap berlaku untuk semua piksel dalam *window* yang berpusat di titik p. Yakni kecepatan vektor (Vx, Vy) harus memenuhi persamaan :

$$Ix(q1)Vx+ Iy(q1)Vy= - It(q)$$

$$Ix(q2)Vx+ Iy(q2)Vy= - It(q1)$$

$$Ix(qn)Vx+ Iy(qn)Vy= - It(q2n)$$

Dimana q1, q2, …….qn adalah piksel dalam *window*, dan Ix Ix)(qi), Iy(qi), It(qi) adalah derivatif parsial dari gambar yang berhubungan dengan posisi x, y, dan waktu t, yang dievaluasi pada titik qi dan pada waktu saat ini. Persamaan ini dapat ditulis dalam bentuk matriks Av = b, dimana :

$$A = \begin{bmatrix} I_x(q_1) & I_y(q_1) \\ \vdots & \vdots \\ I_x(q_n) & I_y(q_n) \end{bmatrix}, v = \begin{bmatrix} V_x \\ V_y \end{bmatrix}, b = \begin{bmatrix} -I_t(q_1) \\ -I_t(q_2) \\ \vdots \\ -I_t(q_{1n}) \end{bmatrix}$$

2.7 *Dashboard car camera* (*Dash Cam*)

Dash Cam atau kamera dashboard adalah kamera yang biasanya ditempatkan di kaca spion tengah untuk secara terus-menerus merekam apa yang terjadi di hadapan kendaraan maupun didalam kendaraan. Kegunaan Dash Cam ini antara lain :

- Menjadi “kotak hitam” apabila terjadi kecelakaan di jalan raya.
- Menjadi bukti apabila terjadi tindak kejahatan atau penyimpangan yang melanggar kesopanan yang dilakukan oleh penumpang.

Untuk bekerja secara otomatis, perangkat ini menggunakan sumber listrik dari mobil, dengan cara mengambil listrik dari pemantik api yang biasanya terdapat pada *dashboard* mobil dan mengaktifkan proses merekam ketika mesin mobil dinyalakan, dan perangkat akan berhenti beroperasi ketika mesin mobil dimatikan. Sekarang ini, semua produk *car camera* menggunakan *memory card* sebagai media penyimpanan video perjalanan. Ketika *memory card* telah terisi penuh, maka *car camera* akan menulis ulang file yang berumur paling lama pada *memory card*. Metode ini disebut *loop recording*. Semakin besar kapasitas *memory card* yang dipergunakan, semakin banyak pula video yang dapat disimpan.

Gambar 2.4. Car Dashboard Camera HD DVR

III. METODE PENELITIAN

3.1 Waktu dan Tempat Penelitian

Penelitian ini mulai dilaksanakan pada bulan Mei 2016 di Laboratorium Teknik Elektronika, Jurusan Teknik Elektro, Universitas Lampung.

3.2 Alat dan Bahan

Alat dan bahan yang digunakan dalam pembuatan tugas akhir ini adalah sebagai berikut :

1. *Dashboard car camera*
2. Stop Kontak
3. Multiport USB *Charger*
4. Perangkat lunak *OpenCV* 2.4.13
5. Perangkat lunak *Visual Studio* Express 2010
6. Kabel USB Tipe mini-B (2.0)
7. Laptop HP 1000

3.3 Spesifikasi Alat

Spesifikasi alat adalah sebagai berikut :

1. Alat ini menggunakan sumber energi yang berasal dari listrik PLN, yang kemudian diubah menjadi *DC* 5 Volt menggunakan *Multiport USB Charger*.
2. *Dashboard car camera* digunakan sebagai pengindra, yang bertugas mengambil citra objek yang ada didepannya. *Dashboard car camera* ini menggunakan masukan *DC* 5V, menggunakan kabel usb 2.0, dan menghasilkan keluaran berupa video dengan format file *AVI*.
3. Hasil keluaran dari *Dashboard car camera* akan secara otomatis disimpan pada SD Card, pada penelitian ini saya menggunakan SD Card dengan kapasitas 16gb.
4. File video dengan format *AVI* pada SD *Card* akan dijadikan sebagai masukan pada pengolahan citra yang akan dilakukan.
5. Pengolahan citra pada penelitian kali ini menggunakan perangkat lunak OpenCV 2.4.10 dan *Visual Studio Express* 2010 pada Laptop HP 1000 sebagai media pemrograman.

3.4 Spesifikasi Sistem

Spesifikasi sistem adalah sebagai berikut :

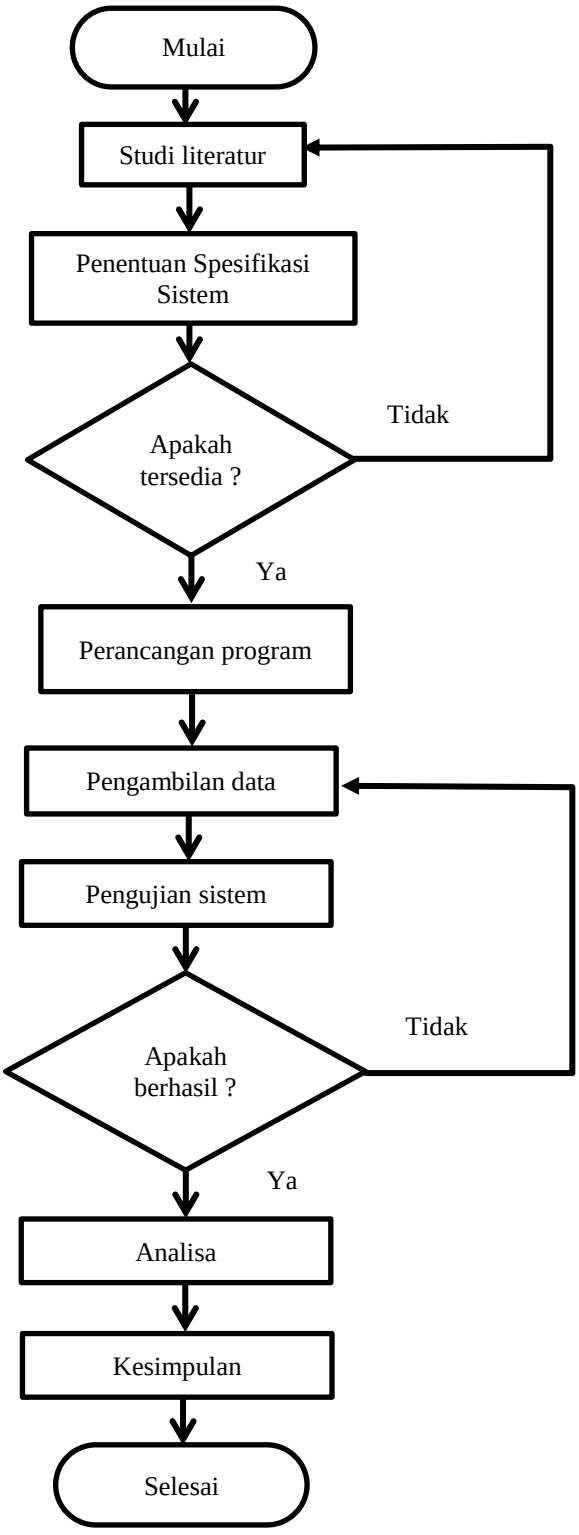
- Mampu mengambil video dari objek di depan dengan devais pengindra *Dashboard car camera*, selanjutnya keluarannya yang berupa file video akan menjadi masukan sistem. Sistem yang akan dibangun menggunakan perangkat lunak OpenCV dan *Visual Studio* sebagai media pemrogramannya. Selanjutnya sistem diharapkan mampu melakukan pendeteksian pola pada dimensi objek yang diamati. Jenis dimensi yang akan diamati adalah dimensi dua dan tiga saja.

3.5 Metode Penelitian

Pada penelitian dan perancangan tugas akhir ini, langkah-langkah kerja yang dilakukan adalah sebagai berikut :

3.5.1. Diagram Alir Penelitian

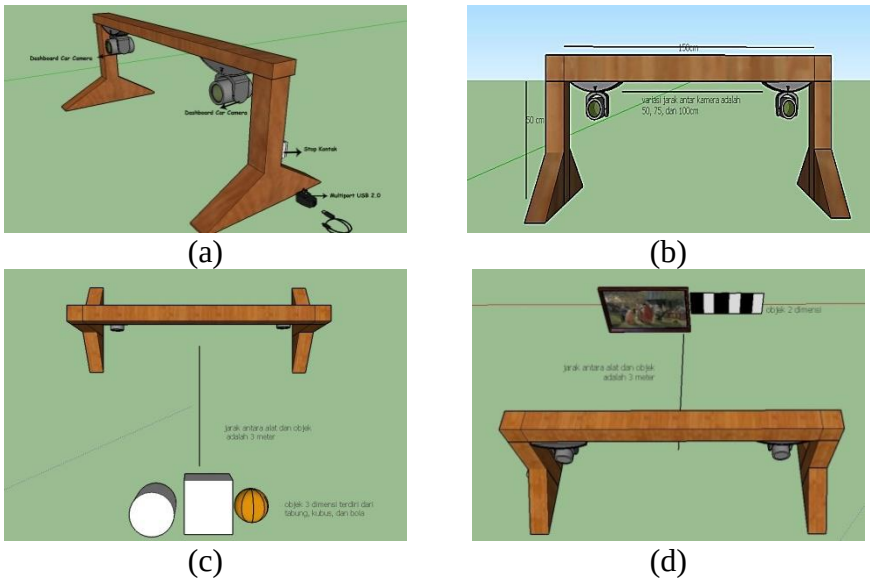
Diagram alir penelitian ini dibuat untuk memperjelas langkah-langkah kerja yang akan dilakukan dalam penelitian, diperlihatkan pada Gambar 3.1. dibawah ini :



Gambar 3.1. Diagram Alir Penelitian

3.5.2. Perancangan Perangkat Keras

Secara umum perangkat keras yang akan digunakan telah dijadikan satu kesatuan yang meliputi, *Dashboard car camera* , Stop kontak, Kabel USB 2.0. Rancangan perangkat keras ini secara keseluruhan dapat dilihat pada Gambar 3.2.



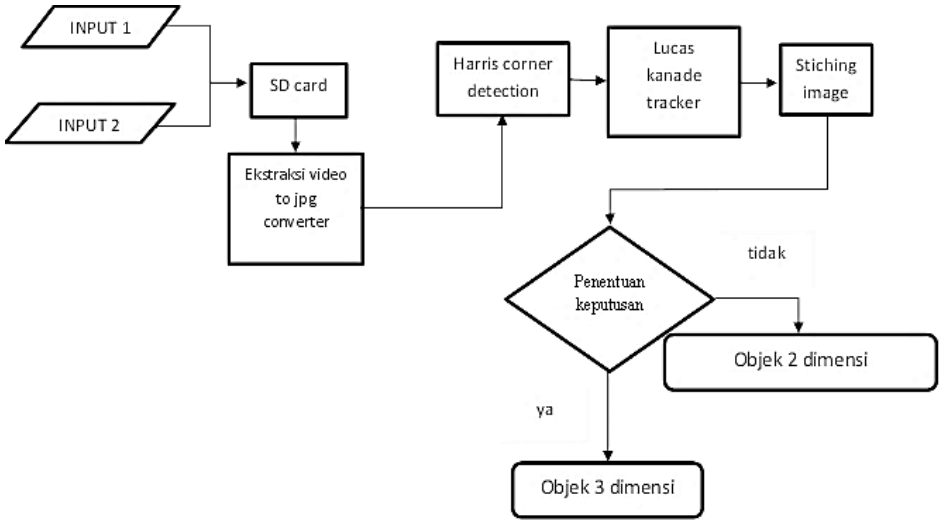
Gambar 3.2. (a) Rancang Bangun Alat Tampak Samping, (b) Detail Rancang Bangun Beserta Ukurannya, (c) Rancang Bangun Sistem dan Obyek 3 Dimensi, (d) Rancang Bangun Sistem dan Obyek 2 Dimensi

3.5.3. Perancangan Program

Perangkat lunak yang digunakan pada penelitian ini yaitu OpenCV 2.4.13. Perangkat lunak ini berfungsi untuk membuat program pada *Visual Studio* 2010. Sistem kerja dari Visual Studio 2010 yakni mengoperasikan komponen komponen yang terdapat pada OpenCV 2.4.13 agar mampu dioperasikan dan diatur sepenuhnya untuk menciptakan sistem yang diinginkan.

3.5.4. Perancangan Model Sistem

Secara keseluruhan sistem dapat dilihat pada Gambar 3.3



Gambar 3.3. Diagram Blok Keseluruhan Sistem

Dari blok diagram terlihat bahwa masukan yang berupa citra berasal dari *dashboard car camera* yang mempunyai keluaran format file video, selanjutnya file tersebut diekstrak untuk dapat dianalisa dimensi objeknya, proses ekstraksi menghasilkan beberapa buah file format *bmp* pada setiap *frame*nya. Setelah proses ekstraksi file dengan format *bmp* hasil keluaran dari *Dash-Cam* tadi diproses menggunakan metode *Harris corner Detection*, metode ini digunakan untuk mendapat titik titik yang kita inginkan pada sebuah citra, kemudian kedua citra yang telah didapat titiknya akan dikorelasikan hingga titik titik pada kedua citra tersebut saling terhubung menggunakan metode *Lucas kanade*.

Selanjutnya setelah dikorelasikan, kedua citra tersebut disatukan kembali guna mengetahui jenis dimensi objek yang diamati apakah objek tersebut 3 dimensi atau 2 dimensi.

3.6. Pengujian Sistem

Uji coba sistem ini dilakukan untuk mengetahui tingkat keberhasilan dari alat yang telah dibuat. Pada tahap uji coba ini keseluruhan perangkat pada yang meliputi perangkat keras seperti rangka penyangga, *Dashboard car camera* , Stop Kontak, akan dicoba untuk dijalankan sesuai dengan prosedur yang telah dibuat, apakah sesuai dengan prosedur atau tidak. Hal ini dilakukan untuk dapat mengetahui sistem secara keseluruhan dapat bekerja dengan baik atau tidak. Pengujian dilakukan dengan menjalankan kamera sebagaimana mestinya, dimana kamera akan mengambil citra obyek didepannya, selanjutnya citra hasil akan diolah pada sistem agar dapat mendeteksi obyek dengan dimensi dua atau tiga dimensi. Pengujian dapat dikatakan berhasil apabila sistem dapat melakukan pendeteksian dimensi obyek sesuai dengan dimensi obyek asli yang diamati dan setelah melewati sistem sesuai dengan keinginan.

V. KESIMPULAN DAN SARAN

5.1. Kesimpulan

Setelah mendapatkan hasil dari metode yang telah diujikan didapatkan kesimpulan sebagai berikut :

1) Telah terealisasi program pendeteksian dimensi obyek menggunakan metode *harris corner* dan *lucas kanade* berbasis citra stereo dengan menggunakan obyek bola, kubus, tabung, dan gambar dua dimensi, dengan variasi jarak antar kamera adalah 10,20, dan 50cm serta jarak antar obyek dan titik tengah kamera adalah 1,5m.
2) Metode yang diusulkan yakni *harris corner* dan *lucas kanade* mampu mendeteksi obyek berdimensi tiga pada obyek bola, kubus, dan tabung dengan jarak antar kamera 10, dan 20cm dengan tingkat presisi rata rata diatas 50%, sedangkan untuk jarak antar kamera 50cm hasil buruk yang didapat pada keseluruhan obyek pengamatan dengan hasil dari nilai presisi rata rata dibawah 20%

5.2. Saran

Adapun saran dari penelitian ini adalah:

1) Untuk pengembangan alat ini selanjutnya perlu diperhatikan komponen pengindra yang benar benar identik agar hasil data yang didapat lebih baik dan presisi.
2) Dari hasil penelitian yang dilakukan, jarak yang disarankan untuk program mampu mendeteksi obyek yang diinginkan antara 10 hingga 20cm dengan komponen pengindra *Dash-Cam* dan dikembangkan pada keadaan sebenarnya, sehingga sistem dapat secara langsung diterapkan pada kehidupan sehari hari.
3) Pada penelitian selanjutnya dapat dikembangkan dengan menggunakan *live streaming*, agar obyek yang diamati dapat secara langsung dideteksi.

DAFTAR PUSTAKA

Hermawan, Fany. (2014). [online]. Tersedia: http://elib.unikom.ac.id/. UNIVERSITAS KOMPUTER INDONESIA. [diakses 2 Agustus 2016]

Hadwi Permata Anggi. (2013). Simulasi Dan Analisis Metode Level Set Untuk Deteksi Kontur Objek 3d Sederhana Berbasis Stereovision. Universitas Telkom

Sany Aji, M. (2015). Deteksi Sudut Pada Gambar 2d Berurutan Dengan Menggunakan Metode Harris/Plessey Corner Detector. Politeknik Negeri Surabaya, Surabaya

Mulyawan, H (2014). Identifikasi Dan Tracking Objek Berbasis *Image Processing* Secara *Real Time*. Politeknik Negeri Surabaya, Surabaya

Anonymous. (2015). Pengantar Pengolahan Citra. Jaka Pramana, Jakarta.324 hlm.

Alhamzi Khaled. (2014). 3D *Object Recognition Based on Image Features*, *International Journal of Computer and Information Technology*, Egypt.

Mursyidah. (2012). Estimasi Gerakan Pada Video Animasi 2d Menggunakan Algoritma Pencocokan Blok (*Block Matching Algorithm*). Seminar Nasional Pengembangan Teknologi Berkelanjutan, Aceh.

Yosuke Igarashi. (2011). 3D *Object Recognition Based on Canonical Angles between Shape Subspaces*. University of Tsukuba, Japan.

Luis A, Alexandre. (2012) *3D Object Recognition using Convolutional Neural Networks with Transfer Learning between Masukan Channels*, Univ Beira Interior, Portugal.

M. Y. Mashor. (2011). *3D Object Recognition Using Multiple Views and Neural Networks*, University Kejuruteraan Utara, Malaysia.

Iryna Gordon. (2010). *3D Object Recognition with Accurate Pose.* University of British, Columbia.

Sumantra Dutta Roy. (2012). *Active 3-D Object Recognition through Next View Planning*, Indian Institute of Technology Bombay, India.

Solihin, Achmad. (2013). Deteksi Pejalan Kaki pada Video dengan Metode Fastest Pedestrian Detector in The West (FPDW). Universitas Gadjah Mada, Yogyakarta.

Arinto, Setyawan FX. (2014). *Detecting Moving Objects From a Video Taken By a Moving Camera Using Sequential Inference Of Background Images*. 19th International Symposium on Artificial Life and Robotics, Japan.

Pujiyanta, Ardi. (2009). Pengenalan Citra Objek Sederhana Dengan Jaringan Saraf Tiruan Metode Perceptron, Universitas Ahmad Dahlan, Yogyakarta.

Rachmawati (2012). Rekonstruksi Obyek Tiga Dimensi Dari Citra Dua Dimensi Menggunakan Epipolar Geometry, Universitas Gadjah Mada, Yogyakarta.

LAMPIRAN

Evaluasi Metode Menggunakan *Recall* Dan *Precision* Pada Obyek Bola

Jarak Antar Kamera	**Bola**		
	Frame Ke	*Recall*	*Precision*
10cm	82	100%	52.6%
	86	100%	60.06%
	91	100%	58.90%
	93	100%	59.10%
	Frame Ke	*Recall*	*Precision*
20cm	3	100%	56.00%
	25	100%	55.60%
	48	100%	59.32%
	65	100%	62.80%
	Frame Ke	*Recall*	*Precision*
50cm	15	66%	51,70%
	19	0%	0%
	23	100%	7.7%
	52	100%	10.08%

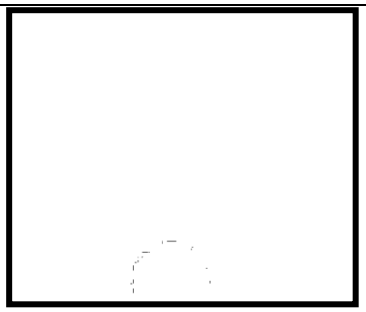
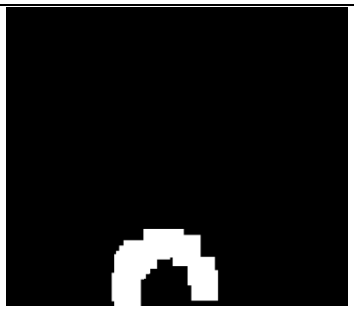
Evaluasi Metode Menggunakan *Recall* Dan *Precision* Pada Obyek Kubus

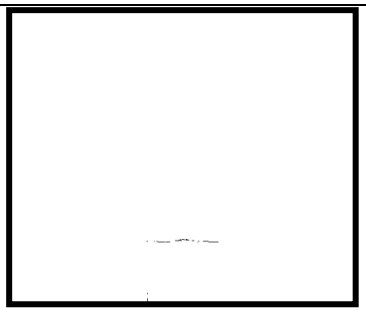
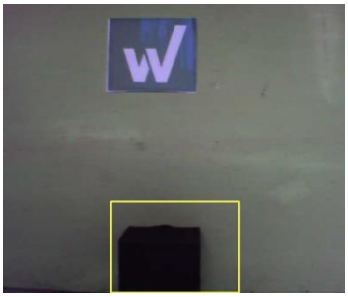
Jarak Antar Kamera	**Kubus**		
	Frame Ke	*Recall*	*Precision*
10cm	3	100%	53,70%
	6	100%	56,40%
	17	100%	61,30%
	93	100%	57,3%
	Frame Ke	*Recall*	*Precision*
20cm	4	100%	51,90%
	5	100%	54,60%
	6	100%	50,10%
	13	100%	50,70%
	Frame Ke	*Recall*	*Precision*
50cm	11	41,2%	32,90%
	15	100%	10,50%
	20	100%	7,17%
	25	100%	6,90%

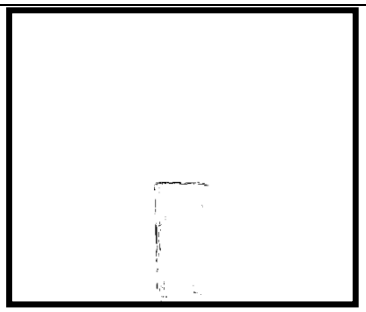
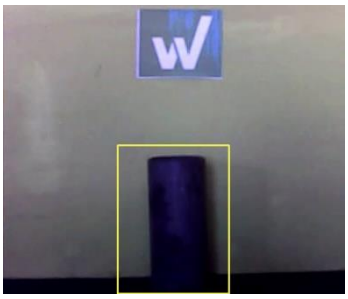
Evaluasi Metode Menggunakan *Recall* Dan *Precision* Pada Obyek Tabung

Jarak Antar Kamera	**Tabung**		
	Frame Ke	*Recall*	*Precision*
10cm	10	100%	63,80%
	12	100%	66,70%
	15	100%	63,50%
	64	100%	55,90%
	Frame Ke	*Recall*	*Precision*
20cm	27	100%	58,70%
	30	100%	57,80%
	36	100%	56,70%
	46	100%	58,80%
	Frame Ke	*Recall*	*Precision*
50cm	2	100%	11,0%
	4	100%	10,7%
	6	100%	12,4%
	8	78%	19,2%

Data Hasil Pendeteksian Sistem

Frame T	Citra *Result*	Citra Morfologi	Penentuan Obyek Bola
82			

Frame T	Citra *Result*	Citra Morfologi	Penentuan Obyek Kubus
4			

Frame T	Citra *Result*	Citra Morfologi	Penentuan Obyek Tabung
10			

“Gambar Awal, dan Format Gray”

```
#include <opencv\cv.h>
#include <opencv\cxcore.h>
#include <opencv\highgui.h>

using namespace cv;
using namespace std;

String inttostr(int input)
{
stringstream ss;
ss << input;
return ss.str();
}
int photocount = 0;
int photocount2 = 0;
int photocount3 = 0;
String imagename;
String imagename2;
String imagename3;

int main()
{
int key = 0;

CvCapture* capture = cvCaptureFromAVI ( "D:/hasil/video
sumber/kotak/50cm/kiri.avi" );
CvCapture* capture2 = cvCaptureFromAVI( "D:/hasil/video
sumber/kotak/50cm/kanan.avi" );
IplImage* frame = cvQueryFrame( capture );


IplImage* frame2 = cvQueryFrame( capture2 );


while ((key = waitKey(30)) != 27)
{


photocount++;
photocount2++;
photocount3++;


frame = cvQueryFrame( capture );
IplImage* gray = cvCreateImage(cvGetSize(frame), 8, 1);
cvCvtColor(frame, gray, CV_BGR2GRAY);
Mat img(frame);
imagename = "D:/hasil/tampilan video awal/kotak/50cm (640x480)/kanan/kanan"
+ inttostr(photocount) + ".bmp";
imwrite(imagename,img);


frame2 = cvQueryFrame( capture2 );
IplImage* gray2 = cvCreateImage(cvGetSize(frame2), 8, 1 );
cvCvtColor(frame2, gray2, CV_BGR2GRAY);
Mat img2(frame2);
```

```
imagename2 = "D:/hasil/tampilan video awal/kotak/50cm (640x480)/kiri/kiri "
+ inttostr(photocount2) + ".bmp";
imwrite(imagename2,img2);

CvSize img_sz = cvGetSize(gray);
IplImage* imgC = cvCreateImage( cvSize( frame->width+frame2->width, frame-
>height), IPL_DEPTH_8U, 3 );
cvZero( imgC );
cvSetImageROI( imgC, cvRect( 0, 0, frame->width, frame->height ) );
cvCopy(frame, imgC);
cvResetImageROI(imgC);
cvSetImageROI( imgC, cvRect(frame2->width, 0, frame2->width+frame->width,
frame2->height) );
cvCopy(frame2, imgC);
cvResetImageROI(imgC);

Mat img3(imgC);
imagename3 = "D:/hasil/tampilan video awal/kotak/50cm
(640x480)/digabung/imgC " + inttostr(photocount3) + ".bmp";
imwrite(imagename3,img3);


cvShowImage("windowname", imgC);

}

}
```

“Gambar Korespondensi, dan Corner”

```
#include <opencv\cv.h>
#include <opencv\cxcore.h>
#include <opencv\highgui.h>
#include <stdio.h>
#include <iostream>
#include <opencv2\highgui\highgui.hpp>
#include <opencv2\imgproc\imgproc.hpp>

using namespace cv;
using namespace std;


int main( int argc, char** argv )
{

const int MAX_CORNERS = 100;
int corner_count=MAX_CORNERS;
int win_size  = 3;
float prev [500]={0};
float curr [500]={0};
float posisi1=0;
float posisi2=0;
int w=0;
int inew=0;
int jnew=0;

char features_found[ MAX_CORNERS];
float feature_errors [MAX_CORNERS];

IplImage* imgA=cvLoadImage("kanan1.bmp",CV_LOAD_IMAGE_GRAYSCALE);
IplImage* imgACol=cvLoadImage("kanan1.bmp",CV_LOAD_IMAGE_COLOR);
IplImage* imgB=cvLoadImage("kiri 1.bmp",CV_LOAD_IMAGE_GRAYSCALE);
IplImage* imgBCol=cvLoadImage("kiri 1.bmp",CV_LOAD_IMAGE_COLOR);
CvSize img_sz = cvGetSize(imgA);
IplImage* varian = cvCreateImage(img_sz, IPL_DEPTH_8U,3);
IplImage* means = cvCreateImage(img_sz, IPL_DEPTH_8U,3);
       cvZero(varian);
cvCopy(imgACol, means);


IplImage* eig_image = cvCreateImage(img_sz, IPL_DEPTH_32F,1);
IplImage* tmp_image = cvCreateImage(img_sz, IPL_DEPTH_32F,1);
IplImage* imgF      = cvCreateImage(img_sz, IPL_DEPTH_8U,1);
IplImage* imgResult = cvCreateImage(img_sz, IPL_DEPTH_8U,3);
IplImage* imgColA   = cvCreateImage(img_sz, IPL_DEPTH_8U,3);
IplImage* back      = cvCreateImage(img_sz, IPL_DEPTH_8U,3);
IplImage* mean1     = cvCreateImage(img_sz, IPL_DEPTH_8U,3);
IplImage* mean2     = cvCreateImage(img_sz, IPL_DEPTH_8U,3);
IplImage* mean      = cvCreateImage(img_sz, IPL_DEPTH_8U,3);
```

```
CvPoint2D32f* cornersA = new CvPoint2D32f[MAX_CORNERS];
CvPoint2D32f* cornersB = new CvPoint2D32f[MAX_CORNERS];

cvGoodFeaturesToTrack(
imgA,
eig_image,
tmp_image,
cornersA,
&corner_count,
0.01,
5.0,
0,
5,
0,
0.04);

cvFindCornerSubPix(
imgA,
cornersA,
corner_count,
cvSize(win_size, win_size),
cvSize(-1,-1),
cvTermCriteria(CV_TERMCRIT_ITER|CV_TERMCRIT_EPS,20,0.03));

IplImage* imgC = cvCreateImage( cvSize( imgA->width+imgB->width, imgA-
>height), IPL_DEPTH_8U, 3 );
cvZero( imgC );
cvSetImageROI( imgC, cvRect( 0, 0, imgA->width, imgA->height ) );
cvCopy(imgACol, imgC); //, stacked, NULL );
cvResetImageROI(imgC);
cvSetImageROI( imgC, cvRect(imgB->width, 0, imgB->width+imgA->width, imgB-
>height) );
cvCopy(imgBCol, imgC); //, stacked, NULL );
cvResetImageROI(imgC);


CvSize pyr_sz = cvSize(imgA->width+8, imgA->height/3);
CvSize temp = cvSize(imgB->width, imgB->height*2);

IplImage* pyrA = cvCreateImage(pyr_sz, IPL_DEPTH_32F,1);
IplImage* pyrB = cvCreateImage(temp, IPL_DEPTH_32F,1);

CvMat *imgFmat = cvCreateMat(imgA->height, imgA->width, CV_64FC1);
CvMat *imgGmat = cvCreateMat(imgA->height, imgA->width, CV_64FC1);
CvMat *imgAColmat = cvCreateMat(imgA->height, imgA->width, CV_8UC3);
CvMat *imgBColmat = cvCreateMat(imgA->height, imgA->width, CV_8UC3);
CvMat *varians = cvCreateMat(imgA->height, imgA->width, CV_8UC3);

cvConvert(imgB, imgFmat);
cvConvert(imgA, imgGmat);
cvConvert(imgACol, imgAColmat);
cvConvert(imgBCol, imgBColmat);

cvCalcOpticalFlowPyrLK(
imgA,
imgB,
pyrA,
pyrB,
cornersA,
cornersB,
```

```
corner_count,
cvSize(win_size, win_size),
5,
features_found,
feature_errors,
cvTermCriteria(CV_TERMCRIT_ITER | CV_TERMCRIT_EPS, 20, .3),
0
);

Mat frame(imgACol);
Mat frame2(imgBCol);
for( int i = 0; i < corner_count; i++ )
{      circle( frame, cornersA[i], 5, Scalar(150), -1);
circle( frame2, cornersB[i], 5, Scalar(125), -1);
}

IplImage copy = frame;
IplImage* frameD = &copy;
IplImage copy2 = frame2;
IplImage* frameD2 = &copy2;
IplImage* corner = cvCreateImage( cvSize( imgA->width+imgB->width, imgA-
>height), IPL_DEPTH_8U, 3 );
cvZero( corner );
cvSetImageROI( corner, cvRect( 0, 0, imgA->width, imgA->height ) );
cvCopy(frameD, corner); //, stacked, NULL );
cvResetImageROI(corner);
cvSetImageROI( corner, cvRect(imgB->width, 0, imgB->width+imgA->width,
imgB->height) );
cvCopy(frameD2, corner); //, stacked, NULL );
cvResetImageROI(corner);



for (int i=0; i<corner_count; i++)
{
if (features_found[i]==0||feature_errors[i]>600)
{
continue;
}

CvPoint p0 = cvPoint(
(cornersA[i].x ),
(cornersA[i].y )
);
CvPoint p1 = cvPoint(
(cornersB[i].x )+imgA->width,
(cornersB[i].y )
);

cvLine(imgC,p0,p1,CV_RGB(255,0,0),1);
w=w+1;


prev[2*w-2]=(cornersA[i].y);       //dibuat satu baris!
prev[2*w-1]=(cornersA[i].x);
curr[2*w-2]=(cornersB[i].y);
curr[2*w-1]=(cornersB[i].x);

}
```

```
CvMat mImg, mWorld;
cvInitMatHeader(&mImg, w, 2, CV_32FC1, prev);
cvInitMatHeader(&mWorld, w, 2, CV_32FC1, curr);
CvMat *h1 = cvCreateMat(3, 3, CV_32FC1);
CvMat *h3 = cvCreateMat(3, 3, CV_32FC1);
int vals[] = { 1, 0, 0, 0, 1, 0, 0, 0, 1 };


if (w>4)
{
cvFindHomography(  &mImg, &mWorld, h1,CV_RANSAC,0.99);

cvmInvert (h1, h3);

}
else  cvInitMatHeader(h3, 3, 3, CV_32FC1, vals);


for(int i=0;i<imgA->height;i++)
{
for(int j=0;j<imgA->width;j++)
{
(float) posisi1= ((((i*(float)cvGetReal2D(h3, 0,
0))+(j*(float)cvGetReal2D(h3, 0, 1))+(float)cvGetReal2D(h3, 0,
2))/((i*(float)cvGetReal2D(h3, 2, 0))+(j*(float)cvGetReal2D(h3, 2,
1))+(float)cvGetReal2D(h3, 2, 2))));
(float) posisi2= ((((i*(float)cvGetReal2D(h3, 1,
0))+(j*(float)cvGetReal2D(h3, 1, 1))+(float)cvGetReal2D(h3, 1,
2))/((i*(float)cvGetReal2D(h3, 2, 0))+(j*(float)cvGetReal2D(h3, 2,
1))+(float)cvGetReal2D(h3, 2, 2))));

//define the position of the pixel
if ((posisi1>=0 && posisi2>=0) && (posisi2<imgA->width-1 && posisi1<imgA-
>height-1))
{
int r=(int) posisi1;
int s=(int) posisi2;
float p=abs(r+0.5-posisi1);
float q=abs(s+0.5-posisi2);


if (r=cvRound(posisi1))
{
inew=r-1;
}
else  inew=r+1;

if (s=cvRound(posisi2))
{
jnew=s-1;
}
else jnew=s+1;

CvScalar ijscal=(cvGet2D(imgAColmat,r,s));
CvScalar inewscal=(cvGet2D(imgAColmat,inew,s));
CvScalar jnewscal=(cvGet2D(imgAColmat,r,jnew));
CvScalar ijnewscal=(cvGet2D(imgAColmat,inew,jnew));
CvScalar mean;
mean.val[0]=((p*q*(ijnewscal.val[0]))+((1-p)*q*inewscal.val[0])+((1-
q)*p*jnewscal.val[0])+((1-p)*(1-q)*ijscal.val[0]));
```

```
mean.val[1]=((p*q*(ijnewscal.val[1]))+((1-p)*q*inewscal.val[1])+((1-
q)*p*jnewscal.val[1])+((1-p)*(1-q)*ijscal.val[1]));
mean.val[2]=((p*q*(ijnewscal.val[2]))+((1-p)*q*inewscal.val[2])+((1-
q)*p*jnewscal.val[2])+((1-p)*(1-q)*ijscal.val[2]));
CvScalar thou;
thou.val[0]=(((ijnewscal.val[0])-mean.val[0])*((ijnewscal.val[0])-
mean.val[0])+(inewscal.val[0]-mean.val[0])*(inewscal.val[0]-
mean.val[0])+(jnewscal.val[0]-mean.val[0])*(jnewscal.val[0]-
mean.val[0])+(ijscal.val[0])-mean.val[0]*(ijscal.val[0])-mean.val[0])/3;
thou.val[1]=(((ijnewscal.val[1])-mean.val[1])*((ijnewscal.val[1])-
mean.val[1])+(inewscal.val[1]-mean.val[1])*(inewscal.val[1]-
mean.val[1])+(jnewscal.val[1]-mean.val[1])*(jnewscal.val[1]-
mean.val[1])+(ijscal.val[1])-mean.val[1]*(ijscal.val[1])-mean.val[1])/3;
thou.val[2]=(((ijnewscal.val[2])-mean.val[2])*((ijnewscal.val[2])-
mean.val[2])+(inewscal.val[2]-mean.val[2])*(inewscal.val[2]-
mean.val[2])+(jnewscal.val[2]-mean.val[2])*(jnewscal.val[2]-
mean.val[2])+(ijscal.val[2])-mean.val[2]*(ijscal.val[2])-mean.val[2])/3;

cvSet2D(varians, i,j,thou);

cvSet2D(imgBColmat, i, j,mean);

}
else
{
(float)posisi1=i;
(float)posisi2=j;
CvScalar mean;
CvScalar BCol=(cvGet2D(imgBColmat,i,j));
mean.val[0]=BCol.val[0];
mean.val[1]=BCol.val[1];
mean.val[2]=BCol.val[2];
}


}


}
cvConvert(imgBColmat, imgResult);

for(int i=1;i<imgA->height-1;i++)
{
for(int j=1;j<imgA->width-1;j++)
{
CvScalar varnew=cvGet2D(varians,i,j);
CvScalar fImg=cvGet2D(imgResult,i,j);
CvScalar fImg1=cvGet2D(imgResult,i-1,j-1);
CvScalar fImg2=cvGet2D(imgResult,i-1,j);
CvScalar fImg3=cvGet2D(imgResult,i-1,j+1);
CvScalar fImg4=cvGet2D(imgResult,i,j-1);
CvScalar fImg5=cvGet2D(imgResult,i,j+1);
CvScalar fImg6=cvGet2D(imgResult,i+1,j-1);
CvScalar fImg7=cvGet2D(imgResult,i+1,j);
CvScalar fImg8=cvGet2D(imgResult,i+1,j+1);

CvScalar meanfImg=cvGet2D(mean1,i,j);
meanfImg.val[0]=(fImg1.val[0]+fImg2.val[0]+fImg3.val[0]+fImg4.val[0]+fImg5.
val[0]+fImg6.val[0]+fImg7.val[0]+fImg8.val[0])/8;
```

```
meanfImg.val[1]=(fImg1.val[1]+fImg2.val[1]+fImg3.val[1]+fImg4.val[1]+fImg5.
val[1]+fImg6.val[1]+fImg7.val[1]+fImg8.val[1])/8;
meanfImg.val[2]=(fImg1.val[2]+fImg2.val[2]+fImg3.val[2]+fImg4.val[2]+fImg5.
val[2]+fImg6.val[2]+fImg7.val[2]+fImg8.val[2])/8;

CvScalar aImg=cvGet2D(imgBCol,i,j);
CvScalar aImg1=cvGet2D(imgBCol,i-1,j-1);
CvScalar aImg2=cvGet2D(imgBCol,i-1,j);
CvScalar aImg3=cvGet2D(imgBCol,i-1,j+1);
CvScalar aImg4=cvGet2D(imgBCol,i,j-1);
CvScalar aImg5=cvGet2D(imgBCol,i,j+1);
CvScalar aImg6=cvGet2D(imgBCol,i+1,j-1);
CvScalar aImg7=cvGet2D(imgBCol,i+1,j);
CvScalar aImg8=cvGet2D(imgBCol,i+1,j+1);

CvScalar meanaImg=cvGet2D(mean2,i,j);
meanaImg.val[0]=(aImg1.val[0]+aImg2.val[0]+aImg3.val[0]+aImg4.val[0]+aImg5.
val[0]+aImg6.val[0]+aImg7.val[0]+aImg8.val[0])/8;
meanaImg.val[1]=(aImg1.val[1]+aImg2.val[1]+aImg3.val[1]+aImg4.val[1]+aImg5.
val[1]+aImg6.val[1]+aImg7.val[1]+aImg8.val[1])/8;
meanaImg.val[2]=(aImg1.val[2]+aImg2.val[2]+aImg3.val[2]+aImg4.val[2]+aImg5.
val[2]+aImg6.val[2]+aImg7.val[2]+aImg8.val[2])/8;

CvScalar XX=cvGet2D(back,i,j);
CvScalar XY;
CvScalar XZ;
CvScalar XW=cvGet2D(mean,i,j);

XY.val[0]=255;
XY.val[1]=255;
XY.val[2]=255;
XX.val[0]=abs(aImg.val[0]-fImg.val[0]);
XX.val[1]=abs(aImg.val[1]-fImg.val[1]);
XX.val[2]=abs(aImg.val[2]-fImg.val[2]);
XZ.val[0]=0;
XZ.val[1]=0;
XZ.val[2]=0;
XW.val[0]=abs(meanaImg.val[0]-meanfImg.val[0]);
XW.val[1]=abs(meanaImg.val[1]-meanfImg.val[1]);
XW.val[2]=abs(meanaImg.val[2]-meanfImg.val[2]);

if
(((XX.val[0]+XX.val[1]+XX.val[2])/(varnew.val[0]+varnew.val[1]+varnew.val[2
]))<1)
cvSet2D(varian,i,j,XZ);

else cvSet2D(varian,i,j,XY);
}
}

cvNamedWindow("ImageHomop",1);
cvNamedWindow("ImageResult",1);
cvNamedWindow("LKpyr_OpticalFlow",1);
cvShowImage("ImageHomop",varian);
cvShowImage("ImageResult",imgResult);
cvShowImage("LKpyr_OpticalFlow",imgC);
cvShowImage("corner",corner);
cvWaitKey(0);
return 0;
}
```

"Gambar Overlap Yang Dihasilkan”

```
#include <opencv\cv.h>
#include <opencv\cxcore.h>
#include <opencv\highgui.h>
#include <iostream>
#include <fstream>
#include <string>

using namespace cv;
using namespace std;
int key;
String inttostr(int input)
{
stringstream ss;
ss << input;
return ss.str();
}
int photocount = 0;
int photocount2 = 0;
int photocount3 = 0;
String imagename;
String imagename2;
String imagename3;
int main()
{

CvCapture* capture = cvCaptureFromAVI ( "D:/hasil/video
sumber/kotak/10cm/kiri.avi" );
CvCapture* capture2 = cvCaptureFromAVI( "D:/hasil/video
sumber/kotak/10cm/kanan.avi" );

IplImage* imgACol = cvQueryFrame( capture );
IplImage* imgBCol = cvQueryFrame( capture2 );

while ((key = waitKey(30)) != 27)
{

        ofstream outFile;
outFile.open("cornerkubus10cm");

photocount++;
photocount2++;
photocount3++;

imgACol = cvQueryFrame( capture );
imgBCol = cvQueryFrame( capture2 );
```

```
cvShowImage("kanan", imgACol);
cvShowImage("kiri", imgBCol);

IplImage* imgA = cvCreateImage(cvGetSize(imgACol), 8, 1);
cvCvtColor(imgACol, imgA, CV_BGR2GRAY);
IplImage* imgB = cvCreateImage(cvGetSize(imgBCol), 8, 1);
cvCvtColor(imgBCol, imgB, CV_BGR2GRAY);

const int MAX_CORNERS = 100;
int count=MAX_CORNERS;
int win_size  = 3;
int w=0;
char features_found[ MAX_CORNERS];
float feature_errors [MAX_CORNERS];
float prev [500]={0};
float curr [500]={0};
int inew=0;
int jnew=0;
float posisi1=0;
float posisi2=0;

CvSize img_sz = cvGetSize(imgA);
IplImage* varian = cvCreateImage(img_sz, IPL_DEPTH_8U,3);
IplImage* means = cvCreateImage(img_sz, IPL_DEPTH_8U,3);
cvCopy(imgACol, means);
IplImage* frame_temp = cvCreateImage(img_sz, IPL_DEPTH_32F,1);
IplImage* frame_eigen = cvCreateImage(img_sz, IPL_DEPTH_32F,1);
IplImage* imgResult = cvCreateImage(img_sz, IPL_DEPTH_8U,3);
IplImage* back      = cvCreateImage(img_sz, IPL_DEPTH_8U,3);
IplImage* mean      = cvCreateImage(img_sz, IPL_DEPTH_8U,3);
IplImage* mean1     = cvCreateImage(img_sz, IPL_DEPTH_8U,3);
//menciptakan gambar back
IplImage* mean2     = cvCreateImage(img_sz, IPL_DEPTH_8U,3);
CvPoint2D32f* corners = new CvPoint2D32f[MAX_CORNERS];
CvPoint2D32f* corners1 = new CvPoint2D32f[MAX_CORNERS];


cvGoodFeaturesToTrack(
imgA,
frame_eigen,
frame_temp,
corners,
&count,
0.01,
5.0,
0,
5,
```

```
0,
0.04
);

cvFindCornerSubPix  (          imgA,
corners,
count,
cvSize(win_size, win_size),
cvSize(-1,-1),
cvTermCriteria(CV_TERMCRIT_ITER|CV_TERMCRIT_EPS,20,0.03));

IplImage* imgC = cvCreateImage( cvSize( imgA->width+imgB->width, imgA-
>height), IPL_DEPTH_8U, 3 );
cvZero( imgC );
cvSetImageROI( imgC, cvRect( 0, 0, imgA->width, imgA->height ) );
cvCopy(imgACol, imgC); //, stacked, NULL );
cvResetImageROI(imgC);
cvSetImageROI( imgC, cvRect(imgB->width, 0, imgB->width+imgA->width,
imgB->height) );
cvCopy(imgBCol, imgC); //, stacked, NULL );
cvResetImageROI(imgC);


CvSize pyr_sz = cvSize(imgA->width+8, imgA->height/3);
CvSize temp = cvSize(imgB->width, imgB->height*2);

IplImage* pyrA = cvCreateImage(pyr_sz, IPL_DEPTH_32F,1);
IplImage* pyrB = cvCreateImage(temp, IPL_DEPTH_32F,1);


CvMat *imgAColmat = cvCreateMat(imgA->height, imgA->width, CV_8UC3);
CvMat *imgBColmat = cvCreateMat(imgA->height, imgA->width, CV_8UC3);
CvMat *varians = cvCreateMat(imgA->height, imgA->width, CV_8UC3);


cvConvert(imgACol, imgAColmat);
cvConvert(imgBCol, imgBColmat);


cvCalcOpticalFlowPyrLK(
imgA,
imgB,
pyrA,
pyrB,
corners,
corners1,
count,
cvSize(win_size, win_size),
```

```
5,
features_found,
feature_errors,
cvTermCriteria(CV_TERMCRIT_ITER | CV_TERMCRIT_EPS, 20, .3),
0
);


Mat frame(imgACol);
Mat frame2(imgBCol);
for( int i = 0; i < count; i++ )
{         circle( frame, corners[i], 5, Scalar(150), -1);
circle( frame2, corners1[i], 5, Scalar(125), -1);
}

IplImage copy = frame;
IplImage* frameD = &copy;
IplImage copy2 = frame2;
IplImage* frameD2 = &copy2;
IplImage* corner = cvCreateImage( cvSize( imgA->width+imgB->width, imgA-
>height), IPL_DEPTH_8U, 3 );
cvZero( corner );
cvSetImageROI( corner, cvRect( 0, 0, imgA->width, imgA->height ) );
cvCopy(frameD, corner); //, stacked, NULL );
cvResetImageROI(corner);
cvSetImageROI( corner, cvRect(imgB->width, 0, imgB->width+imgA->width,
imgB->height) );
cvCopy(frameD2, corner); //, stacked, NULL );
cvResetImageROI(corner);



for(int i=0;i<count;i++)
{  if (features_found[i]==0||feature_errors[i]>600)
{
continue;
}

CvPoint p0 = cvPoint(
cvRound( corners[i].x ),
cvRound( corners[i].y )
);
CvPoint p1 = cvPoint(
cvRound( corners1[i].x )+imgA->width,
cvRound( corners1[i].y )
);
```

```
cvLine(imgC,p0,p1,CV_RGB(255,0,0),1);
w=w+1;
prev[2*w-2]=(corners[i].y);
prev[2*w-1]=(corners[i].x);
curr[2*w-2]=(corners1[i].y);
curr[2*w-1]=(corners1[i].x);

outFile<<" -- Corner kanan ["<<i<<"]
("<<corners[i].x<<","<<corners[i].y<<")"<<endl;
outFile<<" -- Corner kiri ["<<i<<"]
("<<corners1[i].x<<","<<corners1[i].y<<")"<<endl;
}

outFile.close();


CvMat mImg, mWorld;
cvInitMatHeader(&mImg, w, 2, CV_32FC1, prev);
cvInitMatHeader(&mWorld, w, 2, CV_32FC1, curr);
CvMat *h1 = cvCreateMat(3, 3, CV_32FC1);
CvMat *h3 = cvCreateMat(3, 3, CV_32FC1);
int vals[] = { 1, 0, 0, 0, 1, 0, 0, 0, 1 };

if (w>4)
{cvFindHomography(  &mImg, &mWorld, h1,CV_RANSAC,1);
cvInvert (h1, h3);
}
else  cvInitMatHeader(h3, 3, 3, CV_32FC1, vals);


for(int i=0;i<imgA->height;i++)
{
for(int j=0;j<imgA->width;j++)
{
(float) posisi1= ((((i*(float)cvGetReal2D(h3, 0, 0))+(j*(float)cvGetReal2D(h3, 0,
1))+
(float)cvGetReal2D(h3, 0, 2))/((i*(float)cvGetReal2D(h3, 2, 0))+
(j*(float)cvGetReal2D(h3, 2, 1))+(float)cvGetReal2D(h3, 2, 2))));
(float) posisi2= ((((i*(float)cvGetReal2D(h3, 1, 0))+(j*(float)cvGetReal2D(h3, 1,
1))+
(float)cvGetReal2D(h3, 1, 2))/((i*(float)cvGetReal2D(h3, 2, 0))+
(j*(float)cvGetReal2D(h3, 2, 1))+(float)cvGetReal2D(h3, 2, 2))));

if ((posisi1>=0 && posisi2>=0) && (posisi2<imgA->width-1 && posisi1<imgA-
>height-1))
{
int r=(int) posisi1;
int s=(int) posisi2;
```

```
float p=abs(r+0.5-posisi1);
float q=abs(s+0.5-posisi2);

if (r=cvRound(posisi1))
{
inew=r-1;
}
else  inew=r+1;

if (s=cvRound(posisi2))
{
jnew=s-1;
}
else jnew=s+1;


CvScalar ijscal=(cvGet2D(imgAColmat,r,s));
CvScalar inewscal=(cvGet2D(imgAColmat,inew,s));
CvScalar jnewscal=(cvGet2D(imgAColmat,r,jnew));
CvScalar ijnewscal=(cvGet2D(imgAColmat,inew,jnew));
CvScalar mean;
mean.val[0]=((p*q*(ijnewscal.val[0]))+((1-p)*q*inewscal.val[0])+((1-
q)*p*jnewscal.val[0])+((1-p)*(1-q)*ijscal.val[0]));
mean.val[1]=((p*q*(ijnewscal.val[1]))+((1-p)*q*inewscal.val[1])+((1-
q)*p*jnewscal.val[1])+((1-p)*(1-q)*ijscal.val[1]));
mean.val[2]=((p*q*(ijnewscal.val[2]))+((1-p)*q*inewscal.val[2])+((1-
q)*p*jnewscal.val[2])+((1-p)*(1-q)*ijscal.val[2]));
mean.val[0]=ijscal.val[0];
mean.val[1]=ijscal.val[1];
mean.val[2]=ijscal.val[2];
CvScalar thou;
thou.val[0]=(((ijnewscal.val[0])-mean.val[0])*((ijnewscal.val[0])-
mean.val[0])+(inewscal.val[0]-mean.val[0])*(inewscal.val[0]-
mean.val[0])+(jnewscal.val[0]-mean.val[0])*(jnewscal.val[0]-
mean.val[0])+(ijscal.val[0])-mean.val[0]*(ijscal.val[0])-mean.val[0])/3;
thou.val[1]=(((ijnewscal.val[1])-mean.val[1])*((ijnewscal.val[1])-
mean.val[1])+(inewscal.val[1]-mean.val[1])*(inewscal.val[1]-
mean.val[1])+(jnewscal.val[1]-mean.val[1])*(jnewscal.val[1]-
mean.val[1])+(ijscal.val[1])-mean.val[1]*(ijscal.val[1])-mean.val[1])/3;
thou.val[2]=(((ijnewscal.val[2])-mean.val[2])*((ijnewscal.val[2])-
mean.val[2])+(inewscal.val[2]-mean.val[2])*(inewscal.val[2]-
mean.val[2])+(jnewscal.val[2]-mean.val[2])*(jnewscal.val[2]-
mean.val[2])+(ijscal.val[2])-mean.val[2]*(ijscal.val[2])-mean.val[2])/3;

cvSet2D(varians, i,j,thou);
cvSet2D(imgBColmat, i, j,mean);

}
```

```
else
{
(float)posisi1=i;
(float)posisi2=j;
CvScalar mean;
CvScalar BCol=(cvGet2D(imgBColmat,i,j));
CvScalar BColor;
BColor.val[0]=BCol.val[0];
BColor.val[1]=BCol.val[1];
BColor.val[2]=BCol.val[2];
mean.val[0]=BCol.val[0];
mean.val[1]=BCol.val[1];
mean.val[2]=BCol.val[2];
}


}


}
cvConvert(imgBColmat, imgResult);

for(int i=1;i<imgA->height-1;i++)
{
for(int j=1;j<imgA->width-1;j++)
{
CvScalar varnew=cvGet2D(varians,i,j);
CvScalar fImg=cvGet2D(imgResult,i,j);
CvScalar fImg1=cvGet2D(imgResult,i-1,j-1);
CvScalar fImg2=cvGet2D(imgResult,i-1,j);
CvScalar fImg3=cvGet2D(imgResult,i-1,j+1);
CvScalar fImg4=cvGet2D(imgResult,i,j-1);
CvScalar fImg5=cvGet2D(imgResult,i,j+1);
CvScalar fImg6=cvGet2D(imgResult,i+1,j-1);
CvScalar fImg7=cvGet2D(imgResult,i+1,j);
CvScalar fImg8=cvGet2D(imgResult,i+1,j+1);

CvScalar meanfImg=cvGet2D(mean1,i,j);
meanfImg.val[0]=(fImg1.val[0]+fImg2.val[0]+fImg3.val[0]+fImg4.val[0]+fImg5.
val[0]+fImg6.val[0]+fImg7.val[0]+fImg8.val[0])/8;
meanfImg.val[1]=(fImg1.val[1]+fImg2.val[1]+fImg3.val[1]+fImg4.val[1]+fImg5.
val[1]+fImg6.val[1]+fImg7.val[1]+fImg8.val[1])/8;
meanfImg.val[2]=(fImg1.val[2]+fImg2.val[2]+fImg3.val[2]+fImg4.val[2]+fImg5.
val[2]+fImg6.val[2]+fImg7.val[2]+fImg8.val[2])/8;

CvScalar aImg=cvGet2D(imgBCol,i,j);
CvScalar aImg1=cvGet2D(imgBCol,i-1,j-1);
CvScalar aImg2=cvGet2D(imgBCol,i-1,j);
```

```
CvScalar aImg3=cvGet2D(imgBCol,i-1,j+1);
CvScalar aImg4=cvGet2D(imgBCol,i,j-1);
CvScalar aImg5=cvGet2D(imgBCol,i,j+1);
CvScalar aImg6=cvGet2D(imgBCol,i+1,j-1);
CvScalar aImg7=cvGet2D(imgBCol,i+1,j);
CvScalar aImg8=cvGet2D(imgBCol,i+1,j+1);

CvScalar meanaImg=cvGet2D(mean2,i,j);
meanaImg.val[0]=(aImg1.val[0]+aImg2.val[0]+aImg3.val[0]+aImg4.val[0]+aImg
5.val[0]+aImg6.val[0]+aImg7.val[0]+aImg8.val[0])/8;
meanaImg.val[1]=(aImg1.val[1]+aImg2.val[1]+aImg3.val[1]+aImg4.val[1]+aImg
5.val[1]+aImg6.val[1]+aImg7.val[1]+aImg8.val[1])/8;
meanaImg.val[2]=(aImg1.val[2]+aImg2.val[2]+aImg3.val[2]+aImg4.val[2]+aImg
5.val[2]+aImg6.val[2]+aImg7.val[2]+aImg8.val[2])/8;

CvScalar XX=cvGet2D(back,i,j);
CvScalar XY;
CvScalar XZ;
CvScalar XW=cvGet2D(mean,i,j);

XY.val[0]=255;
XY.val[1]=255;
XY.val[2]=255;
XX.val[0]=abs(aImg.val[0]-fImg.val[0]);
XX.val[1]=abs(aImg.val[1]-fImg.val[1]);
XX.val[2]=abs(aImg.val[2]-fImg.val[2]);
XZ.val[0]=0;
XZ.val[1]=0;
XZ.val[2]=0;
XW.val[0]=abs(meanaImg.val[0]-meanfImg.val[0]);
XW.val[1]=abs(meanaImg.val[1]-meanfImg.val[1]);
XW.val[2]=abs(meanaImg.val[2]-meanfImg.val[2]);
if
(((XX.val[0]+XX.val[1]+XX.val[2])/(varnew.val[0]+varnew.val[1]+varnew.val[2
]))<1)
cvSet2D(varian,i,j,XZ);

else cvSet2D(varian,i,j,XY);
}
}
```

```
Mat hasil(varian);
imagename = "D:/hasil/result/kotak/50cm (640x480)/ " + inttostr(photocount) +
".bmp";
//imwrite(imagename,hasil);
cvShowImage("ImageHomop",varian);
//cvShowImage("ImageResult",imgResult);
Mat korespondensi(imgC);
imagename2 = "D:/hasil/koresponden/kotak/10cm (640x480)/koresponden " +
inttostr(photocount2) + ".bmp";
imwrite(imagename2,korespondensi);
//cvShowImage("LKpyr_OpticalFlow",imgC);

Mat point(corner);
imagename3 = "D:/hasil/corner/kotak/10cm (640x480)/corner " +
inttostr(photocount3) + ".bmp";
imwrite(imagename3,point);
//cvShowImage("corner",corner);

printf("Detected Points : %d\n", count);
printf("Corespondence point : %d\n", count);

}

}
```

“Gambar Morfologi”

```
#include <opencv2\imgproc\imgproc.hpp>
#include <opencv2\highgui\highgui.hpp>
#include <stdlib.h>
#include <stdio.h>

using namespace cv;

Mat src, dst;

int morph_elem = 0;
int morph_size = 0;
int morph_operator = 0;
int const max_operator = 4;
int const max_elem = 2;
int const max_kernel_size = 21;

char* window_name = "Morphology Transformations Demo";

void Morphology_Operations( int, void* );
int main( int argc, char** argv )
{
src = imread( "D:/hasil/morfologi/source/tabung/50cm/result 8.bmp");

if( !src.data )
{ return -1; }
namedWindow( window_name, CV_WINDOW_AUTOSIZE );
createTrackbar("Operator:\n 0: Opening - 1: Closing \n 2: Gradient - 3: Top
Hat \n 4: Black Hat", window_name, &morph_operator, max_operator,
Morphology_Operations );
createTrackbar( "Element:\n 0: Rect - 1: Cross - 2: Ellipse", window_name,
&morph_elem, max_elem,
Morphology_Operations );
createTrackbar( "Kernel size:\n 2n +1", window_name,
&morph_size, max_kernel_size,
Morphology_Operations );
Morphology_Operations( 0, 0 );

waitKey(0);
return 0;
}
void Morphology_Operations( int, void* )
{
int operation = morph_operator + 2;

Mat element = getStructuringElement( morph_elem, Size( 2*morph_size + 1,
2*morph_size+1 ), Point( morph_size, morph_size ) );
morphologyEx( src, dst, operation, element );
imshow( window_name, dst );
imwrite("D:/hasil/morfologi/result/tabung/50cm/8.bmp",dst);
}
```

“Gambar Bounding Box”

```
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/imgproc/imgproc.hpp"
#include <iostream>
#include <stdio.h>
#include <stdlib.h>

using namespace cv;
using namespace std;

Mat src; Mat src_gray,src1,src_blur;
int thresh = 1;
int max_thresh = 255;
int main( int argc, char** argv )
{
src = imread("D:/hasil/morfologi/result/bola/50cm/52.bmp",1);
src1 = imread("D:/hasil/morfologi/result/bola/50cm/52'.bmp",1);
cvtColor( src, src_gray, CV_BGR2GRAY );
blur(src_gray, src_blur, Size(50,50));


imshow( "source_window", src );

Mat threshold_output;
Mat cap;
src1.copyTo(cap);
vector<vector<Point> > contours;
vector<Vec4i> hierarchy;
threshold( src_blur, threshold_output, thresh, 255, THRESH_BINARY );
findContours( threshold_output, contours, hierarchy, CV_RETR_TREE,
CV_CHAIN_APPROX_SIMPLE, Point(0, 0) );
vector<vector<Point> > contours_poly( contours.size() );
vector<Rect> boundRect( contours.size() );
vector<Point2f>center( contours.size() );
vector<float>radius( contours.size() );

for( int i = 0; i < contours.size(); i++ )
{ approxPolyDP( Mat(contours[i]), contours_poly[i], 3, true );
boundRect[i] = boundingRect( Mat(contours_poly[i]) );
minEnclosingCircle( (Mat)contours_poly[i], center[i], radius[i] );
}

Mat drawing = Mat::zeros( threshold_output.size(), CV_8UC3 );
for( int i = 0; i< contours.size(); i++ )
{
rectangle( cap, boundRect[i].tl(), boundRect[i].br(), Scalar(80,240,240),
2, 8, 0 );
}
cout<<"Find object: "<<contours.size()<<endl;
imshow( "Contours", cap );
imwrite("D:/hasil/Bounding box/bola/50cm/52.bmp",cap);
waitKey (0);
return 0;
}
```