

***INTERACTIVE WHITEBOARD MENGGUNAKAN PENA LED CAHAYA  
TAMPAK BERBASIS KAMERA SMARTPHONE ANDROID***

**SKRIPSI**

Disusun untuk melengkapi syarat-syarat guna memperoleh gelar Sarjana Sains



**diajukan oleh:**

**MUHAMAD ALDIANSYAH**

**3225130874**

**PROGRAM STUDI FISIKA  
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM  
UNIVERSITAS NEGERI JAKARTA**

**2017**

## ABSTRAK

**Muhamad Aldiansyah.** *Interactive Whiteboard* Menggunakan Pena LED Cahaya Tampak Berbasis Kamera *Smartphone* Android. Skripsi, Jakarta : Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Negeri Jakarta, 2017.

Media pembelajaran semakin berkembang seiring waktu berjalan. Salah satunya adalah *interactive whiteboard*. *Interactive whiteboard* merupakan papan tulis interaktif yang membuat pengguna dapat berinteraksi dengan papan tulis elektronik menggunakan suatu antarmuka pengguna. Namun, kendala untuk penggunaan hal ini adalah biaya *interactive whiteboard* tinggi pada sistem *interactive whiteboard* komersil yang harus dibebankan ke sekolah. Berdasar hal tersebut dibuat *interactive whiteboard* menggunakan pena LED cahaya tampak berbasis kamera *smartphone* Android. Perekaman video cahaya LED pada tiga posisi berbeda dengan resolusi kamera 320 x 240, 480 x 360, dan 960 x 720 piksel diproses menggunakan metode *color detection*. Hasil yang didapat adalah penggunaan resolusi kamera 640 x 480 dan 960 x 720 piksel memiliki tingkat presisi lebih tinggi dengan standar deviasi 2.47, dan 2.92 piksel serta lebih responsif dengan waktu 8.29 dan 7.34 detik untuk menulis garis lurus sepanjang 75 cm. Selain itu, cara memegang *pointer* LED dapat memengaruhi hasil deteksi karena cahaya dari LED akan dapat lebih menyebar pada layar proyeksi LCD. *Interactive whiteboard* yang dibuat pun telah diujikan pada aplikasi simulasi fisika dan dapat berjalan cukup baik. Dari hasil tersebut disimpulkan bahwa *interactive whiteboard* menggunakan pena LED cahaya tampak berbasis kamera *smartphone* dapat dibuat dan digunakan dengan jarak antara proyektor LCD dan dinding  $\pm 1.2$  meter dan kamera *smartphone* yang ter-install aplikasi IP Webcam dengan kualitas kamera yang memadai. Selain itu, proyektor perlu diatur dengan tingkat kecerahan yang rendah.

**Kata Kunci:** *Interactive Whiteboard, LED, Smartphone, Android.*

## ABSTRACT

**Muhamad Aldiansyah.** Interactive Whiteboard Using LED Visible Light Pen Based Android Smartphone Camera. Thesis, Jakarta : Mathematics and Natural Science Faculty, State University of Jakarta, 2017.

*Media learning has evolved over time. One of them is interactive whiteboard. The interactive whiteboard is a big interactive screen that allow user to interact with an electronic whiteboard using user interface. But, cost of an interactive whiteboard on a commercial interactive whiteboard that should be charged to school is expensive. Based on it, interactive whiteboard using LED visible-light pen based android smartphone camera is made. LED light is recorded at three different positions with 320 x 240, 480 x 360, and 960 x 720 pixel resolution using color detection method. The results obtained with 640 x 480 and 960 x 720 pixels camera resolution has a higher level of precision with standar deviation 2.47 and 2.92, and more responsive with 8.29 and 7.34 seconds to write a 75cm line . In addition, how to hold LED pointer can affect the detection results because the light from the LED will be more diffuse on the LCD projection screen. The interactive whiteboard has also been tested in physics simulation applications and can run quite well. From this conclusion the interactive whiteboards using LED visible light pens based on camera smartphones can be made and used with the distance between the LCD projector and the wall  $\pm 1.2$  meters and the smartphone camera installed an IP webcam application with adequate camera quality. In addition, the projector must be set with a low brightness level.*

**Keywords:** *Interactive Whiteboard, LED, Smartphone, Android.*

## KATA PENGANTAR

*Bismillahirrahmaanirrahiim.* Puji dan syukur penulis panjatkan kehadiran Allah SWT yang telah senantiasa memberikan rahmat dan nikmat-Nya sehingga skripsi yang berjudul “***Interactive Whiteboard Menggunakan Pena LED Cahaya Tampak Berbasis Kamera Smartphone Android***” dapat terselesaikan dengan baik.

Penulis menyadari bahwa dalam penulisan skripsi ini tak lepas dari adanya bimbingan, bantuan, dan kerjasama dari berbagai pihak. Oleh karenanya, penulis ingin mengucapkan terima kasih kepada :

1. Dr.rer.nat. Bambang Heru Iswanto, M.Si. selaku Dosen Pembimbing I yang telah memberikan bimbingan, motivasi, nasehat, saran, dan kritik kepada penulis.
2. Dr. Widyaningrum Indrasari, M. Si selaku Dosen Pembimbing II dan Ketua Program Studi Fisika Universitas Negeri Jakarta yang telah memberikan bimbingan, motivasi, saran, dan kritik kepada penulis.
3. Dr. Esmar Budi, M. T selaku Penguji Ahli Sidang Skripsi yang telah memberikan saran dan kritik kepada penulis.
4. Orang tua penulis yang telah memberikan doa, nasehat, dukungan baik moril maupun materil kepada penulis.
5. Luthfi Chandra Aliem, Nadya Hidayatie, serta Fisika UNJ 2013 sebagai sahabat saya dalam memberikan ide, motivasi, saran, nasehat, dan sekaligus doa kepada penulis.
6. Rhazafais Suthawa, Romy Khadafi, Nandra Meylana, Taufik Abdullah, Muhammad Idris, dan Lia Iskandar sebagai sahabat yang telah memberikan motivasi kepada penulis.
7. *The Beatles, Oasis, Avenged Sevenfold, Linkin Park, My Chemical Romance, One Ok Rock, dan Bring Me The Horizon* yang telah memberikan semangat secara tidak langsung melalui musik yang dibuat.
8. Serta semua pihak yang telah membantu, yang tidak dapat disebutkan satu persatu.

Demikian beberapa ucapan terima kasih untuk memulai pembuatan proposal skripsi ini. Penulis menyadari bahwa proposal skripsi ini masih banyak kekurangannya baik dalam penyajian materi dan teknik penulisan yang ada nantinya, dapat dijadikan bahan evaluasi kedepan, sehingga proposal ini dapat lebih baik dan lebih inovatif serta dapat dikembangkan oleh penggiat sains yang lain. Untuk itu, segala saran dan kritik sangat penulis harapkan sebagai proses pembelajaran untuk masa yang akan datang.

Jakarta, Agustus 2017

Muhamad Aldiansyah

## DAFTAR ISI

ABSTRAK.....	i
ABSTRACT.....	ii
KATA PENGANTAR .....	iii
DAFTAR ISI.....	v
DAFTAR GAMBAR .....	vii
DAFTAR TABEL.....	ix
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang .....	1
1.2 Identifikasi Masalah .....	4
1.3 Pembatasan Masalah .....	4
1.4 Perumusan Masalah.....	5
1.5 Ruang Lingkup Penelitian.....	5
1.6 Tujuan Penelitian.....	5
1.7 Manfaat Penelitian.....	6
BAB II KAJIAN PUSTAKA.....	7
2.1 Interactive Whiteboard .....	7
2.2 Computer Vision .....	8
2.3 Video Processing.....	9
2.3.1 Gamma.....	9
2.3.2 Gaussian Blur.....	10
2.3.3 Warna.....	12
2.3.4 Operasi Morfologi.....	14
2.4 OpenCV.....	15
2.5 PyAutoGUI.....	15
2.6 Arduino.....	16
BAB III METODOLOGI PENELITIAN .....	18
3.1 Tujuan Operasional .....	18
3.2 Tempat dan Waktu Penelitian .....	18
3.3 Alat dan Bahan Penelitian .....	18
3.4 Metode Penelitian.....	19
3.5 Prosedur Penelitian.....	19

3.5.1	Perancangan Perangkat Keras .....	20
3.5.2.	Perancangan Perangkat Lunak .....	21
3.5.3	Perancangan Pengujian .....	23
3.6	Blok Diagram Sistem .....	25
3.7	Skema Alat .....	25
3.8	Diagram Alir Penelitian .....	26
BAB IV HASIL DAN PEMBAHASAN .....		28
4.1	Pra Eksperimen .....	28
4.1.1	Persiapan Pengujian Interactive Whiteboard .....	28
4.1.2	Setting Alat Interactive Whiteboard .....	29
4.1.3	Kalibrasi Layar Proyeksi LCD dengan Layar Desktop Komputer .	29
4.2	Presisi Deteksi Pointer LED pada Kamera .....	32
4.2.1	Persiapan Eksperimen .....	32
4.2.2	Hasil Eksperimen .....	33
4.3	Presisi Deteksi Pointer Pada Layar Proyeksi LCD .....	37
4.3.1	Persiapan Eksperimen .....	37
4.3.2	Hasil Eksperimen .....	38
4.4	Penggunaan Kamera yang Berbeda.....	41
4.5	Interaksi dengan Aplikasi Simulasi Fisika .....	42
BAB V KESIMPULAN.....		46
5.1	Kesimpulan.....	46
5.2	Saran.....	47
DAFTAR PUSTAKA .....		48
LAMPIRAN.....		50

## DAFTAR GAMBAR

Gambar 1.	Skematik Diagram DigitalDesk [4] .....	7
Gambar 2.	Digitaldesk [4] .....	8
Gambar 3.	Kernel blur dalam Gaussian Blur [9].....	11
Gambar 4.	Plot respon sel kerucut terhadap panjang gelombang [17].....	12
Gambar 5.	Skematik ruang warna RGB [17] .....	13
Gambar 6.	Ruang warna HSV [15] .....	14
Gambar 7.	Arduino UNO [19].....	17
Gambar 8.	Blok diagram perancangan perangkat keras .....	21
Gambar 9.	Blok diagram interactive whiteboard.....	22
Gambar 10.	Blok diagram persiapan pengujian tingkat presisi.....	23
Gambar 11.	Blok diagram sistem .....	25
Gambar 12.	Rancang bangun pena LED cahaya tampak .....	25
Gambar 13.	Rancang bangun interactive whiteboard.....	26
Gambar 14.	Diagram alir penelitian .....	27
Gambar 15.	Setting alat Interactive whiteboard .....	29
Gambar 16.	Persiapan kalibrasi pada layar desktop komputer.....	30
Gambar 17.	Kalibrasi pada layar desktop komputer telah dilakukan.....	30
Gambar 18.	Deteksi pena LED move pada (a) hasil rekaman kamera, (b) hasil region of interest cropping, dan (c) masking image region of interest dengan resolusi kamera 320 x 240 di salah satu frame .....	31
Gambar 19.	Hasil Deteksi Pointer pada Layar Komputer dengan Resolusi Kamera 320 x 240 piksel.....	34
Gambar 20.	Hasil Deteksi Pointer pada Layar Komputer dengan Resolusi Kamera 640 x 480 piksel.....	34
Gambar 21.	Hasil Deteksi Pointer pada Layar Komputer dengan Resolusi Kamera 640 x 480 piksel.....	35
Gambar 22.	Hasil Deteksi Pointer LED pada eksperimen ke-3 pada (a) (b) resolusi kamera 640 x 480 piksel dan pada (c) (d) resolusi kamera 320 x 240 .....	36



Gambar 23. Deteksi pena LED pada (a) hasil region of interest cropping kamera, dan (b) masking image region of interest dengan resolusi kamera 320 x 240 di salah satu frame .....	37
Gambar 24. Perbedaan jarak antara lokasi pena LED ditunjukkan simbol titik dan hasil deteksi ditunjukkan simbol positif .....	39
Gambar 25. Hasil menulis 5 garis lurus dengan menggunakan pena LED .....	40
Gambar 26. Hasil menulis garis lurus untuk menguji delay waktu pada (a) resolusi kamera 320 x 240 piksel (b) resolusi kamera 640 x 480 piksel, dan (c) resolusi kamera 960 x 720 piksel.....	40
Gambar 27. Hasil tulisan dalam program Microsoft Paint menggunakan pointer LED.....	<b>Error! Bookmark not defined.</b>
Gambar 28. Hasil Menulis di Microsoft Paint Menggunakan Kamera Samsung J5 Prime.....	41
Gambar 29. Program simulasi fisika Algoodo dijalankan dengan pena LED interactive whiteboard. (a) Lingkaran kecil diklik, (b) dipindahkan, (c) kemudian dilepas, hingga (d) menggelinding dan menumbuk lingkaran besar.....	42
Gambar 30. Program simulasi fisika Algoodo dijalankan dengan pena LED interactive whiteboard. (a) Lingkaran diklik, (b) ditarik ke bawah (c) kemudian dilepas hingga pegas bekerja dan lingkaran bergerak naik turun.....	43
Gambar 31. Program simulasi fisika Algoodo dijalankan dengan pena LED interactive whiteboard. (a) Lingkaran diklik, (b) ditarik ke kanan (c) kemudian dilepas hingga lingkaran berayun ke kanan dan kiri.....	44

## DAFTAR TABEL

Tabel 1. Tabel deskripsi Arduino UNO .....	17
Tabel 2. Karakteristik Ruang Warna HSV untuk Image Masking .....	28
Tabel 3. Tingkat Presisi Deteksi Pointer LED pada Kamera.....	35
Tabel 4. Data hasil nilai kesalahan deteksi lokasi Pena LED .....	38
Tabel 5. Hasil nilai kesalahan absolut pendeteksian lokasi pena LED.....	39
Tabel 6. Nilai kesalahan pendeteksian menulis garis lurus menggunakan pena LED .....	39
Tabel 7. Waktu yang diperlukan pointer LED untuk menulis garis lurus .....	41

# BAB I

## PENDAHULUAN

### 1.1 Latar Belakang

Pembelajaran merupakan proses interaksi dan komunikasi peserta didik dengan pendidik dan sumber belajar pada suatu lingkungan belajar [1]. Media pembelajaran semakin berkembang seiring waktu berjalan. Salah satunya adalah *interactive whiteboard*. *Interactive whiteboard* merupakan papan tulis interaktif yang membuat pengguna dapat berinteraksi dengan papan tulis elektronik menggunakan suatu antarmuka pengguna. Namun, kendala untuk penggunaan hal ini adalah biaya *interactive whiteboard* tinggi pada sistem *interactive whiteboard* komersil yang harus dibebankan ke sekolah.

Dalam pengoperasiannya, *interactive whiteboard* dapat menggunakan teknologi layar sentuh yang sensitif terhadap tekanan jari atau teknologi yang memanfaatkan sensor kamera, pena LED, proyektor LCD, dan program pengolahan citra digital [2]. Pada tahun 2008 Johnny Chung Lee menginisiasi pembuatan *interactive whiteboard* menggunakan *wii remote* dan pena inframerah dengan harga yang relatif terjangkau dibanding *interactive whiteboard* yang sensitif terhadap tekanan jari [3]. Pada penelitian yang dilakukan Johnny Chung Lee tersebut, gelombang inframerah di koordinat tertentu dari pena inframerah dapat ditangkap oleh *wii remote* yang berfungsi sebagai sensor. Data berupa posisi pada suatu koordinat dari *wii remote* lalu dikirim ke komputer melalui *bluetooth*. Kemudian data tersebut diproses untuk ditampilkan pada dinding melalui proyektor. Hasil yang diperlihatkan di kanal *youtube* milik Johnny Lee menunjukkan keberhasilan pertama memanipulasi *game controller* sebagai dasar membuat *interactive whiteboard*.

Pada tahun 2010, Fariz Darmawan merancang *interactive whiteboard* menggunakan aplikasi *wii remote* dengan pendekatan *high touch design process*. Dalam penelitiannya, Fariz Darmawan menganalisa bahwa gerakan pena inframerah sering tidak terlihat kamera sehingga pergerakannya kurang sesuai dan menjadi sulit dioperasikan. Hal ini bisa disebabkan oleh sinyal dari inframerah yang

digunakan kurang kuat. Berdasar hasil analisisnya, ia kemudian membuat alat yang permanen dengan menambahkan fungsi klik kanan dan desain pena inframerah yang nyaman menggunakan tombol *momentary switch*. Dalam pengujiannya, pengajar yang mencoba alat tersebut memberikan respon setuju pada alatnya dalam meningkatkan ketertarikan, meningkatkan pembelajaran, dan kepuasan. Selain itu, para peserta belajar yang diuji rata-rata memberikan respon antusias meski ada beberapa yang tidak puas [1].

Pada tahun 2012, Andre Yusdianto dkk membuat aplikasi *interactive whiteboard* untuk mendukung pembelajaran menggunakan *controlling game* yaitu *wii remote* serta input berupa sinyal inframerah. Andre yusdianto menjelaskan bahwa papan tulis memiliki ukuran tertentu yang membatasi pendidik dalam menyampaikan materi sehingga materi yang telah disampaikan sebelumnya akan dihapus oleh pendidik karena keterbatasan ruang pada papan tulis dan materi pengajaran yang ditulis di media tidak bisa direkam sehingga penyampaian materi tidak bisa ditangkap dengan baik oleh peserta didik. Dari aplikasi yang telah dibuat dalam penelitian tersebut bisa menyimpan hasil pengajaran yang telah ditulis di layar. [4]

Pada tahun 2012, Soares dkk membuat *interactive whiteboard* menggunakan kamera *webcam* dengan *IR band-pass filter* sebagai sensor dan input berupa LED inframerah. Dalam penelitiannya, Soares merekam beberapa video dengan kondisi berbeda di antaranya adalah mengklik pena inframerah dengan dan tanpa *noise* berupa gangguan cahaya kemudian menguji efisiensi algoritma tersebut dengan meninjau waktu pemrosesan. Hasil menunjukkan bahwa *noise* tersebut dapat mengganggu penggunaan dan mempengaruhi efisiensi algoritma [5].

Di tahun 2012, Zhang dkk membuat *interactive whiteboard* menggunakan *kinect*. *Kinect* berfungsi sebagai sensor yang dapat menangkap data gerak secara 3D. Data dari *kinect* kemudian dikirim ke komputer melalui kabel USB. Data tersebut kemudian diolah komputer menggunakan *CUDA platform* serta *OpenCV* untuk menambah kecepatan memproses gambar. Selain itu, komputer yang digunakan memiliki spesifikasi RAM berukuran 2GB dan kartu grafis GeForce GT 240M. Sistem penangkapan citra pada *kinect* menangkap citra dengan resolusi 640

x 480. Hasil menunjukkan interaksi dapat ditampilkan pada dinding melalui proyektor dengan pengenalan akurasi tertinggi 99.9% pada ukuran proyeksi 0.4 x 0.3 m. Namun, tingkat akurasi pengenalan lokasi pointer semakin berkurang ketika ukuran proyeksi semakin luas. Tingkat akurasi terendah bernilai 90.1% pada ukuran proyeksi 3.2 x 2.4 m. Selain itu, Zhang dkk menemukan bahwa system ini rentan terhadap interferensi inframerah yang kuat dan secara signifikan mengurangi akurasi sistem pemosisian [6].

Pada tahun 2013, Jan Podmajersky membuat *interactive whiteboard* menggunakan pena LED pada cahaya tampak. Ia mengujikan alat dan programnya tersebut pada 9 orang. Secara umum, para *tester* terkesan dengan *interactive whiteboard* tersebut. Namun terdapat kendala seperti *double clicks* yang tidak sensitif, kadang tertekan huruf dobel ketika menulis dengan *software keyboard*, ada tester yang kesulitan menulis karena badannya menghalangi cahaya dari pena LED, dan terdapat ketidakakuratan dalam penggunaan aplikasi *microsoft paint* [7].

Pada tahun 2014, Wei-Kai Liou dan Chun-Yen Chang membuat sistem *interactive whiteboard* menggunakan *webcam* dan cahaya laser, atau dapat disebut *laser-driven interactive system* (LaDIS), berwarna merah sebagai sinyal input. Dalam pengujiannya kepada pengajar dan siswa, nilai SSI (*Student and Teacher Satisfaction Index*) secara keseluruhan mendapatkan nilai di atas 90% untuk kepuasan, keinginan untuk menggunakan, interaksi, pembelajaran, efek, interaksi dengan guru, dan lebih baik dibanding alat tradisional [8].

Setelah menelaah beberapa penelitian di atas, maka akan dibangun *Interactive Whiteboard* Menggunakan Pena LED Cahaya Tampak Berbasis Kamera *Smartphone*. Pada *interactive whiteboard* ini, sensor yang digunakan berupa kamera *smartphone* berbasis android yang dihubungkan ke laptop melalui *wireless local area network* menggunakan aplikasi android IP *webcam*. Perangkat input yang digunakan berupa pena LED cahaya tampak berwarna hijau. Pemilihan warna hijau ini bertujuan supaya warna dapat terbedakan dengan warna kulit manusia yang kadang tertangkap oleh kamera cenderung mendekati warna merah atau biru. Dengan menggunakan LED warna hijau, tingkat kesalahan pendeteksian pena LED menggunakan program *computer vision* dimungkinkan dapat dikurangi. Selain itu,

program arduino dibuat supaya dua *push button* memiliki fungsi yang berbeda, yaitu: menggerakkan pointer dan *men-drag* pointer di desktop komputer. Ada pun penelitian ini bertujuan untuk membuat *interactive whiteboard* dengan tingkat presisi pendeteksian pena interaktif yang cukup tinggi, mobilitas yang memadai, dan sebagai alat interaksi interaktif untuk menjalankan program simulasi fisika di komputer.

## **1.2 Identifikasi Masalah**

Berdasarkan latar belakang masalah yang telah disampaikan, permasalahan-permasalahan yang dapat diidentifikasi adalah sebagai berikut:

1. Bagaimana tingkat efektivitas penggunaan *interactive whiteboard* menggunakan pena LED cahaya tampak berbasis kamera *smartphone*?
2. Bagaimana tingkat akurasi pendeteksian pena LED untuk *interactive whiteboard* menggunakan pena LED cahaya tampak berbasis kamera *smartphone*?
3. Bagaimana tingkat presisi pendeteksian pena LED untuk *interactive whiteboard* menggunakan pena LED cahaya tampak berbasis kamera *smartphone*?
4. Bagaimana pengaruh gangguan cahaya dari luar terhadap *interactive whiteboard* menggunakan pena LED cahaya tampak berbasis kamera *smartphone*?
5. Bagaimana pengaruh resolusi kamera terhadap *interactive whiteboard* menggunakan pena LED cahaya tampak berbasis kamera *smartphone*?

## **1.3 Pembatasan Masalah**

Pada penelitian ini permasalahan akan dibatasi pada pembuatan program *interactive whiteboard* menggunakan kamera *smartphone* Android samsung J5 Prime, arduino uno, dan pena LED cahaya tampak berwarna hijau.

#### 1.4 Perumusan Masalah

Perumusan masalah dalam penelitian ini adalah: “Seberapa efektif penggunaan *interactive whiteboard* menggunakan pena LED cahaya tampak berwarna hijau berbasis kamera *smartphone android*?”

#### 1.5 Ruang Lingkup Penelitian

Agar pembahasan dapat dilakukan dengan lebih cermat, maka ruang lingkup yang akan dibahas pada penelitian ini adalah:

1. Perangkat input sistem *interactive whiteboard* berupa cahaya LED berwarna hijau.
2. Sensor berupa kamera *smartphone* Android Samsung J5 Prime plus berbasis Android yang terhubung dengan laptop melalui *wireless local area network*.
3. Metode *pre-processing* pada program yang digunakan adalah *gamma correction* dan *gaussian blur*, metode pendektasian posisi pena LED yang digunakan adalah metode pengenalan warna, dan metode *post-processing* yang digunakan adalah *dilation* dan *erosion*.
4. Perangkat *output* sistem *interactive whiteboard* berupa proyektor LCD.
5. Pustaka program yang digunakan adalah *OpenCV* dan *PyAutoGUI* untuk bahasa pemrograman Python.
6. Arduino uno digunakan supaya *push button* pena LED saklar pendeteksian pada program Python dapat dibuat untuk dua kondisi, yaitu: menggerakkan *pointer* dan *men-drag pointer* di desktop.

#### 1.6 Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah untuk merancang *interactive whiteboard* menggunakan pena LED cahaya tampak berwarna hijau berbasis kamera *smartphone* Android dengan tingkat presisi pendeteksian pena interaktif yang cukup tinggi, mobilitas yang memadai, dan sebagai alat interkasi interaktif untuk menjalankan program simulasi fisika di komputer

## **1.7 Manfaat Penelitian**

Manfaat yang dapat diambil dari penelitian ini adalah:

1. Pengguna dapat memanfaatkan *interactive whiteboard* menggunakan pena LED cahaya tampak berbasis kamera *smartphone* sebagai kegiatan belajar mengajar secara interaktif.
2. *Interactive whiteboard* dapat digunakan sebagai alat interaksi interaktif untuk simulasi program fisika.
3. Sebagai bahan perbandingan untuk disempurnakan pada penelitian berikutnya.



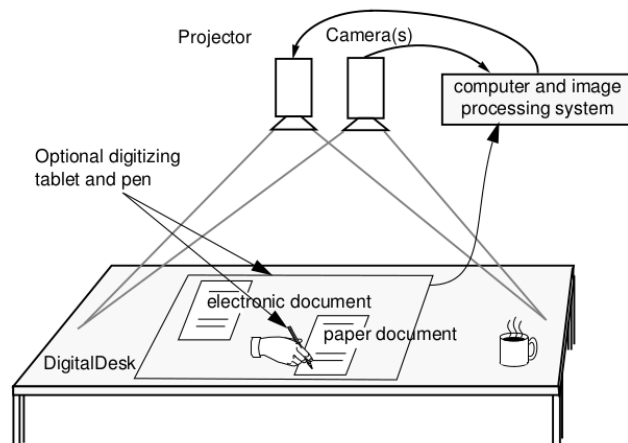
## BAB II

### KAJIAN PUSTAKA

#### 2.1 *Interactive Whiteboard*

*Interactive Whiteboard* dapat dibagi menjadi dua jenis, yaitu dengan berdasar teknologi layar yang sensitif terhadap sentuhan dan teknologi yang berdasar *machine vision*. Teknologi *machine vision* pada awal-awal diturunkan dari *Digital Desk* [4].

*DigitalDesk* adalah meja tulis fisik di mana pengguna dapat menumpuk kertas-kertas, menggunakan pensil dan spidol, dan membiarkan gelas kopi di atasnya, tapi meja tulis ini diperkaya untuk memberikan beberapa karakteristik dari *workstation* elektronik. Suatu tampilan komputer diproyeksikan di atas meja tulis, dan kamera video ditempatkan mengarah ke bawah pada meja tulis dengan melalui sistem pemrosesan citra yang dapat mengerti apa yang pengguna lakukan. (Lihat gambar 1 dan 2).



**Gambar 1.** Skematik Diagram *DigitalDesk* [4]

Satu tujuan dari *digitaldesk* adalah untuk melampaui yang disebut manipulasi langsung dengan *mouse* dan menjelajahi kemungkinan interaksi langsung menggunakan jari. *Digitaldesk* menyatukan kertas dan dokumen elektronik tidak hanya dengan menambahkan sifat elektronik pada dokumen kertas, tapi juga menambahkan sifat fisik ke dokumen elektronik. Kita dapat menggunakan

pena dan jari untuk berinteraksi langsung dengan kertas pada meja tulis kita, jadi kita pun dapat berinteraksi dengan dokumen pada *digitaldesk* dengan cara yang sama [4].



**Gambar 2.** *Digitaldesk* [4]

## 2.2 *Computer Vision*

*Computer vision* adalah bagian dari ilmu pengetahuan komputer yang memiliki tujuan untuk membuat komputer dapat meniru kemampuan penglihatan manusia dengan cara memproses dan memahami citra digital [9]. Pemrosesan citra digital memiliki tiga tingkat dalam implementasinya, yaitu: *low-level*, *mid-level*, dan *high-level*. *Low-level* mempelajari kajian seperti pengurangan *noise* citra, mempertajam citra, peningkatan kualitas citra, dan lain-lain. *Low-level* secara umum memiliki input dan output berupa citra digital. *Mid-level* mengkaji implementasi seperti untuk deteksi fitur citra, estimasi gerakan, rekonstruksi objek 3D, segmentasi citra, dan lain-lain. *Mid-level* secara umum memiliki input berupa citra digital dan output berupa fitur yang diekstrak. Sedangkan *high-level*, dalam hal ini termasuk ke dalam *computer vision*, memiliki kajian yang merupakan bagian dari bidang kecerdasan buatan, yaitu membuat komputer dapat melakukan tugas-tugas tertentu yang biasanya dilakukan oleh sistem penglihatan manusia dalam menginterpretasikan suatu objek seperti pengenalan karakter optik, pemantauan, deteksi wajah, dan lain-lain [10][11].

Pada dasarnya, dalam suatu citra digital terdapat kumpulan piksel di baris dan kolom berukuran tetap yang bernilai antara 0 dan 255 yang merepresentasikan kecerahan pada suatu titik piksel [11].

### 2.3 *Video Processing*

Video adalah kumpulan dari citra bergerak yang berurut terhadap waktu. Jumlah bingkai per detik untuk citra supaya mata manusia tidak merasakan kedipan pergantian citra diperlukan lebih dari 50 bingkai per detik [10]. Dengan demikian maka dalam implementasinya pemrosesan video digital dapat dikatakan sebagai pemrosesan citra digital bergerak yang berurut terhadap waktu.

#### 2.3.1 *Gamma*

Pada awal pengembangan televisi hitam-putih, fosfor dalam tabung sinar katoda digunakan untuk menampilkan respon sinyal televisi secara *non-linear* dari input tegangannya. Hubungan antara tegangan (V) dan hasil kecerahan (B) dikarakterisasi oleh sebuah angka yang disebut *gamma* ( $\gamma$ ), dengan persamaan sebagai berikut dengan gamma sekitar 2.2,

$$B = V^\gamma \quad (2.1)$$

Untuk mengimbangi efek di atas, elektronika pada kamera televisi memetakan penangkapan *luminance*, yaitu kecerahan per satuan luas dari arah tertentu, dalam simbol Y melalui *inverse gamma*,

$$Y^I = Y^{\frac{1}{\gamma}} \quad (2.2)$$

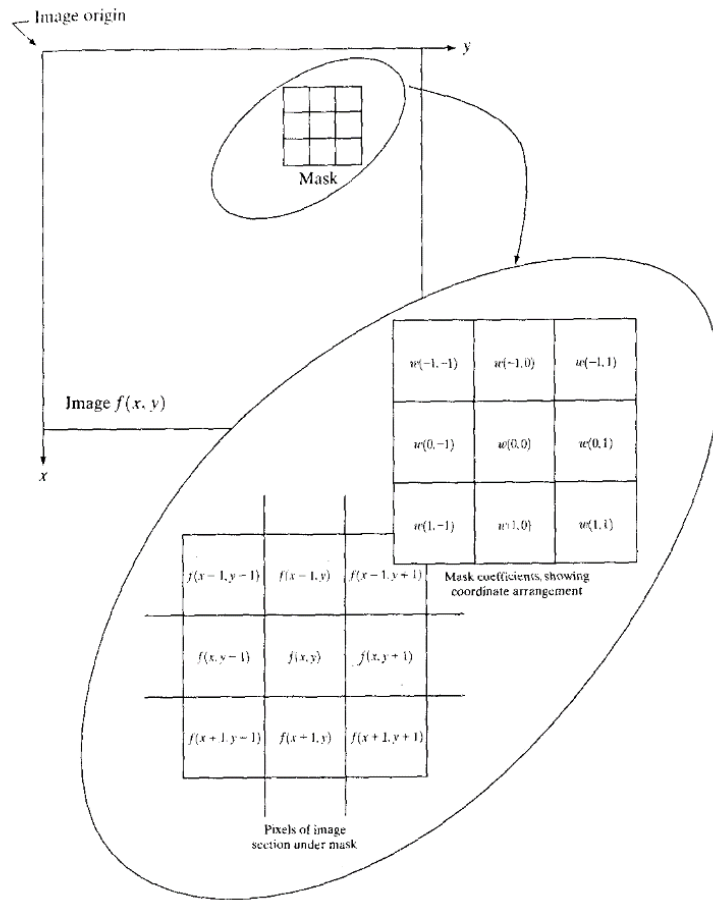
dengan nilai tipikal  $\frac{1}{\gamma} = 0.45$  [13]

### 2.3.2 Gaussian Blur

Suatu citra digital sering terbentuk dengan struktur ideal bersama *noise* yang ingin dihilangkan. Penghalusan citra untuk menghilangkan *noise* dapat dilakukan dengan mengkonvolusikan suatu citra digital dengan suatu kernel yang dapat juga disebut dengan *mask* [11]. Secara umum, fungsi untuk memfilter suatu citra digital dengan suatu kernel adalah sebagai berikut:

$$g(x, y) = \sum_{s=-a}^a \sum_{r=-b}^b w(s, t) f(x + s, y + t) \quad (2.3)$$

Di mana  $a = \frac{(m-1)}{2}$  dan  $b = \frac{(n-1)}{2}$  dengan nilai  $m$  adalah jumlah baris koefisien *mask* dan  $n$  adalah jumlah kolom koefisien *mask*. Selain itu,  $w(s, t)$  adalah kernel yang digunakan dan  $f(x + s, y + t)$  adalah citra yang dioperasikan dengan gaussian blur. Sehingga *array* untuk *mask* berukuran  $m \times n$ . Filter tersebut diterapkan untuk nilai  $x = 0, 1, 2, \dots, M-1$  dan nilai  $y = 0, 1, 2, \dots, N-1$  dengan nilai  $M$  adalah jumlah baris piksel citra dan nilai  $N$  adalah jumlah kolom citra. Sehingga *array* untuk citra yang difilter berukuran  $M \times N$ .  $g(x, y)$  adalah hasil dari penghalusan citra digital. Pengaplikasian suatu *mask* pada citra digital seperti ditunjukkan gambar berikut [9].



**Gambar 3.** Kernel blur dalam Gaussian Blur [9]

Untuk *gaussian blur*, nilai koefisien  $m \times n$  pada *kernel* dihitung dengan menggunakan fungsi gauss [14]:

$$G_{\sigma, \mu_x, \mu_y}(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x-\mu_x)^2}{2\sigma^2}} e^{-\frac{(y-\mu_y)^2}{2\sigma^2}} \quad (2.4)$$

Di mana  $\mu_x$  dan  $\mu_y$  adalah nilai rata-rata yang bernilai 0.  $\sigma$  adalah standar deviasi. Sedangkan untuk  $x$  dan  $y$  adalah komponen bernilai diskrit dari  $-\sigma$  hingga  $\sigma$ .

Dalam penerapan *opencv*, jika nilai  $\sigma$  tidak didefinisikan dalam penggunaan *gaussian blur*, maka nilai  $\sigma$  dapat dihitung untuk komponen  $x$  dan  $y$  sebagai berikut:

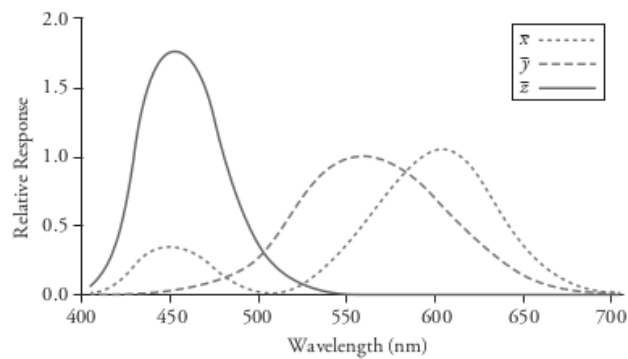
$$\sigma_x = \left(\frac{n_x}{2} - 1\right) 0.30 + 0.80, n_x \quad (2.5)$$

$$\sigma_y = \left(\frac{n_y}{2} - 1\right) 0.30 + 0.80, n_y \quad (2.6)$$

Di mana,  $n_x$  dan  $n_y$  adalah jumlah baris dan kolom untuk kernel yang digunakan.  $\sigma_x$  dan  $\sigma_y$  adalah nilai standar deviasi. [16]

### 2.3.3 Warna

Pada sistem optik mata manusia terdapat 100-120 juta sel batang dan 7-8 juta sel kerucut dalam retina yang dapat membentuk citra pada permukaan retina. Kedua sel tersebut merupakan reseptor yang mengeluarkan sinyal listrik ketika cahaya mengenainya. Ada tiga tipe sel kerucut yang sangat sensitif terhadap panjang gelombang yang pendek, sedang, dan panjang. Untuk sel kerucut panjang gelombang pendek yang disebut *S-cones* memiliki puncak pada 420 nm, untuk sel kerucut panjang gelombang sedang yang disebut *M-cones* memiliki puncak pada 534 nm, dan untuk sel kerucut panjang gelombang panjang yang disebut *L-cones* memiliki puncak pada 564 nm.

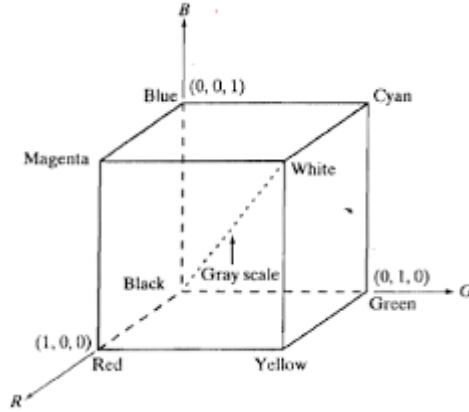


**Gambar 4.** Plot respon sel kerucut terhadap panjang gelombang [17]

### Ruang warna RGB

Model warna RGB terinspirasi dari penglihatan manusia. Model warna RGB merupakan warna aditif yaitu warna merah, hijau, dan biru yang dapat digabungkan untuk memproduksi berbagai macam warna. Model warna ini diterapkan pada

perangkat yang dapat menangkap dan memancarkan cahaya seperti kamera, proyektor video, LCD/LED TV dan monitor komputer, dan layar telepon genggam.



**Gambar 5.** Skematik ruang warna RGB [17]

### Ruang warna HSV

Ruang warna HSV adalah ruang warna yang bersifat non-linear [17]. Dalam ruang warna HSV, ide dasarnya adalah piksel warna terbentuk dari *value* ( $V$ ), yang merupakan intensitas, *saturation* ( $S$ ), yang merupakan jumlah warna, dan *hue* ( $H$ ), yang merupakan warna itu sendiri [12]. Untuk mentransformasikan dari ruang warna RGB ke HSV, dapat digunakan persamaan sebagai berikut,

$$H_1 = \cos^{-1} \left\{ \frac{\frac{1}{2} [(R - B) + (G - B)]}{\sqrt{(R - G)^2 + (R - B)(G - B)}} \right\} \quad (2.7)$$

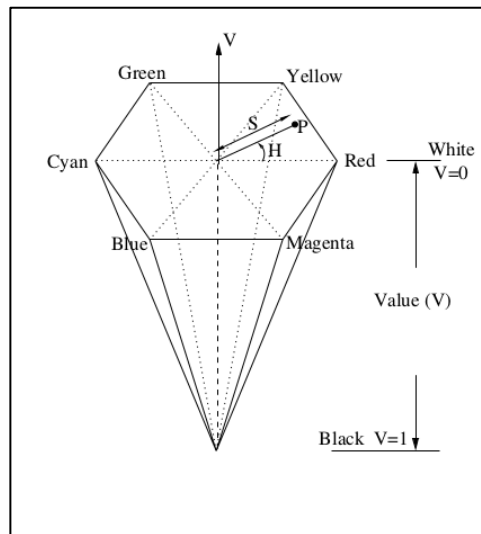
$$H = H_1, \text{ if } B \leq G \quad (2.8)$$

$$H = 360^\circ - H_1, B > G \quad (2.9)$$

$$S = \frac{\max(R, G, B) - \min(R, G, B)}{\max(R, G, B)} \quad (2.10)$$

$$V = \frac{\max(R, G, B)}{255} \quad (2.11)$$

Di mana  $R$ ,  $G$ , dan  $B$  merupakan nilai dari suatu piksel yang merepresentasikan intensitas berada pada rentang 0-255 [15].  $H$  adalah *Hue*,  $S$  adalah *Saturation*, dan  $V$  adalah *Value*.



**Gambar 6.** Ruang warna HSV [15]

### 2.3.4 Operasi Morfologi

Morfologi berarti bentuk dan struktur dari suatu objek, atau susunan dan hubungan timbal-balik antara bagian-bagian objek. Operasi dasar untuk operasi morfologi adalah seperti erosi dan dilasi. Erosi adalah proses di mana tiap piksel pada suatu citra akan dicocokkan dengan suatu kernel dan pada pola tertentu akan dihapus. Sedangkan dilasi adalah proses di mana tiap piksel pada suatu citra akan dicocokkan dengan suatu kernel dan pada pola tertentu akan diperluas. Salah satu penerapan erosi dan dilasi adalah pada citra biner [12].

Untuk citra biner, notasi matematika untuk operasi dilasi adalah sebagai berikut,

$$C = A \oplus B \quad (2.12)$$

Dimana  $\oplus$  menyatakan operasi morfologi dilasi. A adalah citra yang dioperasikan, dan B adalah kernel untuk mengoperasikan erosi pada citra A. C adalah hasil citra.

$$C = A \ominus B \quad (2.13)$$



Dimana  $\ominus$  menyatakan operasi morfologi erosi. A adalah citra yang dioperasikan, dan B adalah kernel untuk mengoperasikan erosi pada citra A. C adalah hasil citra.

## 2.4 OpenCV

OpenCV adalah pustaka *computer vision* yang bersifat *open source*. Pada awalnya pustaka ini dimiliki Intel hingga kemudian beberapa universitas bekerja sama dalam pengembangannya. Bagaimana pun, penggunaan pustaka ini terbatas hanya pada sebagian kecil orang yang melakukan riset dalam bidang *computer vision* di universitas tertentu. Pustaka ini membolehkan mereka untuk membuat aplikasi berkualitas tinggi, dan akhirnya banyak dan lebih banyak orang bekerja sama pada pustaka ini untuk menerbitkan publikasi. Pada awalnya pustaka ini diimplementasikan dalam bahasa pemrograman C, hingga pada versi 2.0 C++ menjadi bahasa pemrograman utama. Selain itu, OpenCV dapat juga dipakai dalam bahasa pemrograman berbeda, misalnya Phyton, Ruby, dan lainnya.

OpenCV dirilis pada bulan Januari 1999, hingga kemudian dikembangkan terus menerus secara berkelanjutan. Pustaka ini menyediakan banyak implementasi algoritma pengolahan citra. Bagaimana pun, pustaka OpenCV sesuai juga untuk implementasi berbeda, misalnya uuntuk pengenalan musik, di mana teknik *vision recognition* diterapkan untuk citra spektrogram suara.

Meskipun sebagai pustaka yang *open source*, pustaka ini dapat digunakan dalam produk komersial tanpa kewajiban untuk memuat kode program. Kebijakan lisensinya sedikit bebas, tapi ini memberikan keuntungan, perusahaan menginginkan untuk membuat produk yang lebih baik dan mereka bekerja sama dalam pengembangan pustaka OpenCV. OpenCV digunakan dalam banyak proyek industri medis, deteksi keamanan, dan lainnya [8].

## 2.5 PyAutoGUI

Tujuan dari PyAutoGUI adalah untuk menyediakan modul lintas-platform Python untuk otomatisasi untuk manusia. Antarmuka pemrograman aplikasi ini didesain supaya sesederhana mungkin. Misalnya untuk menggerakkan mouse ke tengah layar untuk Windows, OS X, dan Linux menggunakan kode program:

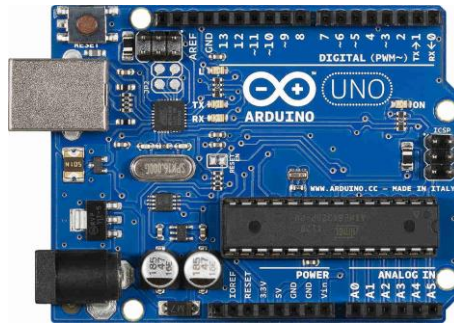
```
>>> import pyautogui
>>> screenWidth, screenHeight = pyautogui.size()
>>> pyautogui.moveTo(screenWidth / 2, screenHeight / 2)
```

PyAutoGUI dapat mensimulasikan gerakan mouse, mengklik mouse, memindahkan kursor mouse, menekan tombol *keyboard*, menahan tombol *keyboard*. Pada Windows, PyAutoGUI tidak memiliki dependensi, untuk OS X, PyAutoGUI memerlukan PyObjC terinstal untuk AppKit dan Quartz dengan nama modul *pyobjc-core* dan *pyobjc* pada PyPI, dan untuk linux, PyAutoGUI memerlukan modul *python-xlib* atau *python3-Xlib* terinstal. [18]

## 2.6 Arduino

Arduino adalah sebuah *board* sistem mikrokontroler yang bersifat *open source*. Di dalam -rangkain *board* arduino terdapat mikrokontroler AVR seri ATmega 328 yang merupakan produk dari Atmel [19]. Arduino menyediakan 20 pin I/O, yang terdiri dari 6 pin input analog dan 14 pin digital input/output. Enam pin analog tersebut dapat difungsikan sebagai output digital jika diperlukan output digital tambahan selain 14 pin yang sudah tersedia. Dalam *board* pin digital diberi keterangan 0-13, sehingga untuk menggunakan pin analog menjadi output digital, pin analog yang pada keterangan board 0-5 kita ubah menjadi pin 14-19.

Sifat *open source* arduino juga banyak memberikan keuntungan tersendiri dalam menggunakan *board* ini, karena komponen yang dipakai tidak hanya tergantung pada satu merk, tetapi memungkinkan pengguna dapat memakai semua komponen yang dijual. Bahasa pemrograman arduino merupakan bahasa C yang sudah disederhanakan *syntax* bahasanya sehingga mempermudah dalam mempelajari dan mendalami mikrokontroler.



**Gambar 7.** Arduino UNO [19]

**Tabel 1.** Tabel deskripsi Arduino UNO

Mikrokontroler	Atmega 328
Tegangan Pengoperasian	5V
Tegangan input yang disarankan	7-12V
Batas tegangan input	6-20V
Jumlah pin I/O digital	14 pin digital (6 diantaranya menyediakan keluaran PWM)
Jumlah pin input analog	6 pin
Arus DC tiap pin I/O	40Ma
Arus DC untuk pin 3,3V	50Ma
<i>Flash Memmory</i>	32 kb (Atmega 328) sekitar 0,5 kb digunakan oleh bootloader
SRAM	2 kb
EPR0M	1 kb
<i>Clock speed</i>	16 MHz

Untuk membuat program Arduino dan meng-*upload* ke dalam *board* arduino diperlukan *software* Arduino IDE. IDE (*Integrated Development Environment*) adalah sebuah *software* yang berperan untuk menulis program, meng-*compile* menjadi kode biner, dan meng-*upload* ke dalam memori mikrokontroler.

## **BAB III**

### **METODOLOGI PENELITIAN**

#### **3.1 Tujuan Operasional**

Tujuan operasional dalam penelitian ini di antaranya adalah sebagai berikut:

1. Membuat program *computer vision* yang dapat digunakan sebagai *interactive whiteboard*.
2. Membuat pena LED cahaya tampak berwarna hijau sebagai alat interaksi pengguna dan laptop.
3. Membuat program arduino yang dapat membedakan *input* menggerakkan pointer desktop dan *input* *men-drag* pointer desktop melalui *push button*.
4. Membuat sistem *interactive whiteboard* dengan kamera *smartphone* Android, proyektor LCD, pena LED, Arduino, serta laptop yang berisi program *computer vision*.
5. Memperoleh data pendeteksian posisi cahaya LED oleh pemrosesan program *computer vision* dari citra yang ditangkap menggunakan kamera *smartphone*.

#### **3.2 Tempat dan Waktu Penelitian**

Penelitian ini dilakukan di Laboratorium Instrumentasi Fisika UNJ pada bulan April – Juli 2017.

#### **3.3 Alat dan Bahan Penelitian**

Alat dan bahan yang digunakan dalam penelitian ini berupa perangkat keras dan perangkat lunak.

Perangkat keras yang digunakan antara lain:

1. Laptop dan Arduino Uno
2. Kamera *Smartphone*
3. LED berwarna hijau
4. Proyektor LCD dan Kabel VGA
5. Kabel

6. *Push button* sebanyak dua buah
7. Wadah spidol bekas dan Penggaris

Perangkat lunak yang digunakan antara lain:

1. Python, *OpenCV*, dan *PyautoGUI*
2. *IP Webcam*
3. *Arduino IDE*

### **3.4 Metode Penelitian**

Metode yang digunakan dalam penelitian ini adalah metode *research and development*. Penelitian ini mengembangkan sistem *interactive whiteboard* berbasis kamera menggunakan kamera *smartphone* dan pena LED cahaya tampak berwarna hijau.

Pada *interactive whiteboard* ini perlu dilakukan kalibrasi untuk menyesuaikan ukuran layar LCD laptop dengan ukuran proyeksi *interactive whiteboard*. Setelah dikalibrasi, kamera dapat digunakan. Dengan menggunakan aplikasi android *IP webcam*, hasil perekaman menggunakan kamera *smartphone* akan dikirim ke laptop melalui *wireless local area network* untuk diproses menggunakan *computer vision* dengan pustaka *opencv*. Koordinat pendeteksian lokasi pena LED pada citra akan disesuaikan dengan koordinat layar LCD laptop menggunakan pustaka *PyAutoGUI*.

### **3.5 Prosedur Penelitian**

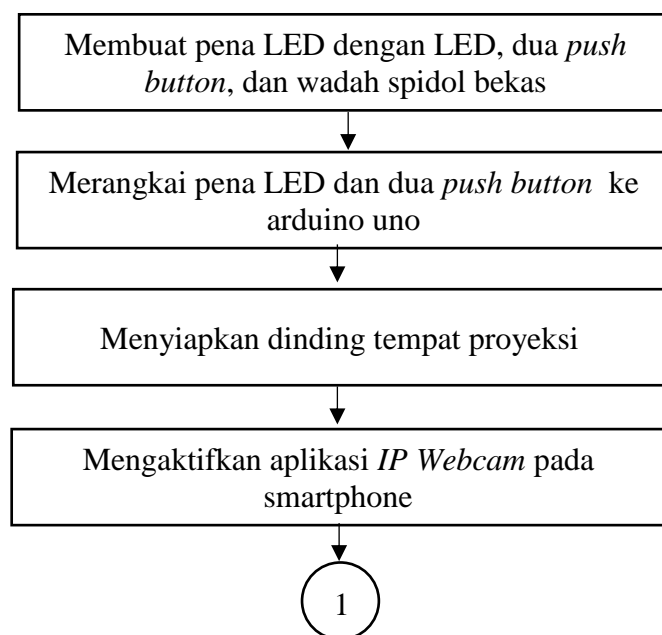
1. Studi literatur mengenai *interactive whiteboard*, *computer vision*, *image processing*, dan pustaka pemrograman yang diperlukan dalam pembuatan *software*.
2. Preparasi perangkat lunak dan perangkat keras yang dibutuhkan.
3. Membuat program *computer vision* menggunakan pustaka *opencv* dan *pyautogui* untuk *interactive whiteboard* berbasis kamera *smartphone*.
4. Membuat program arduino untuk dua kondisi *push button*.

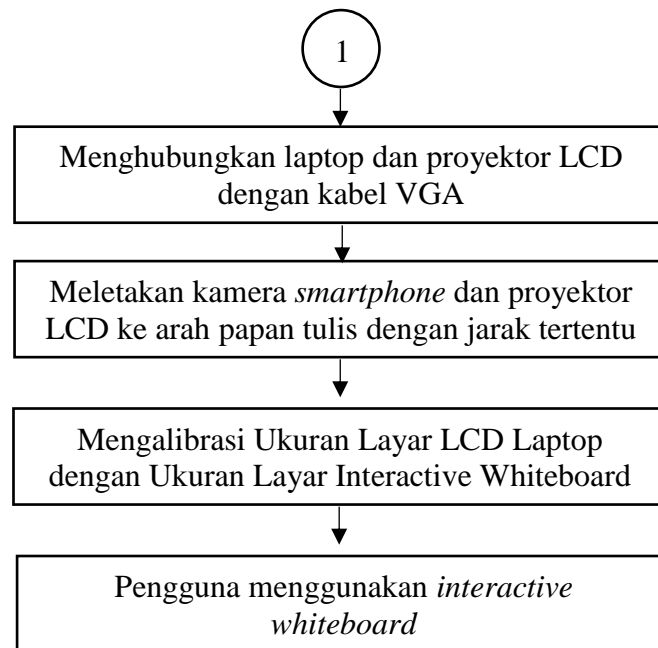
5. Pengambilan data dan pengujian presisi pengenalan posisi pena LED cahaya tampak dengan variasi jarak kamera terhadap pena LED dan variasi gangguan cahaya.

### 3.5.1 Perancangan Perangkat Keras

Pena LED dibuat dengan merangkai LED, dua *push button*, dan kabel pada wadah spidol bekas ke Arduino. Ketika *push button* pertama ditekan maka pena LED akan mengeluarkan cahaya dan memberikan informasi berupa angka 1 pada *serial monitor* kemudian program di *python* mendeteksi cahaya pena LED dan memindahkan lokasi *pointer* di *desktop* ke tempat sesuai kalibrasi. Hal ini berlaku juga untuk *push button* kedua, tetapi untuk *push button* kedua akan memberi informasi berupa angka 2 untuk *men-drag* lokasi *pointer* di *desktop* sesuai kalibrasi. Selain pena LED, perangkat keras seperti laptop, *smartphone*, proyektor LCD, kabel VGA, dan *whiteboard* dirangkai dengan mengarahkan proyektor LCD dan kamera *smartphone* ke *whiteboard*. Setelah sistem *interactive whiteboard* dibuat, pengujian tingkat presisi pengenalan posisi pointer LED dapat dilakukan.

Blok diagram untuk perancangan perangkat keras dapat dilihat pada Gambar 9, rancang bangun pena LED dapat dilihat pada Gambar 13, dan rancang bangun *interactive whiteboard* dapat dilihat pada Gambar 14.

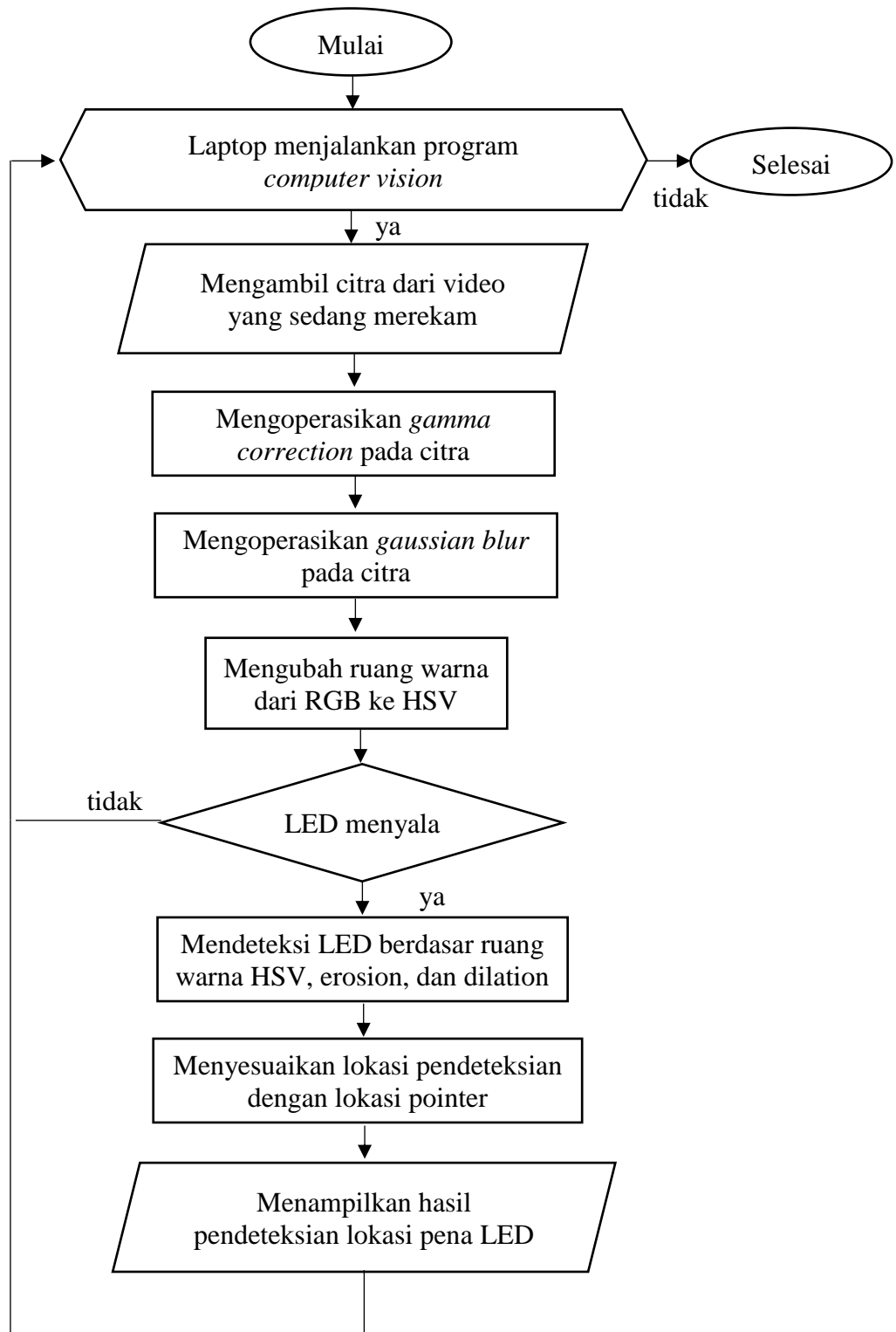




**Gambar 8.** Blok diagram perancangan perangkat keras

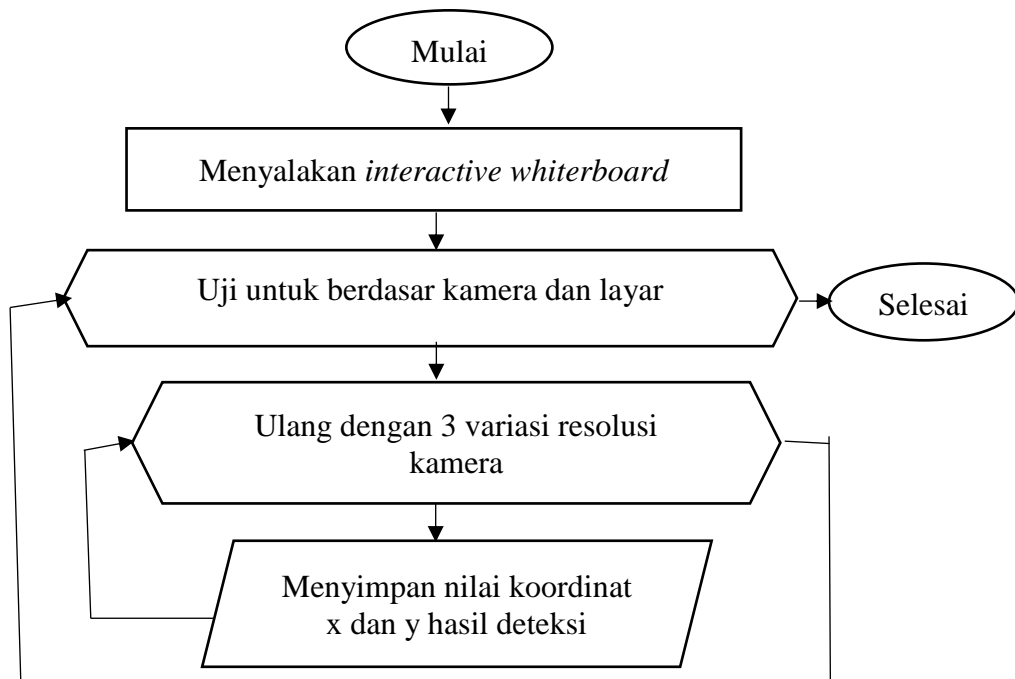
### 3.5.2. Perancangan Perangkat Lunak

Algoritma program *computer vision* untuk mengenali posisi pena LED berisi perintah untuk mengambil citra yang ditangkap oleh kamera *smartphone*. Kemudian program akan melakukan operasi *gaussian blur*, dan mengubah ruang warna dari RGB ke HSV. Ketika pengguna menekan salah satu *push button* hingga LED menyala, maka program akan mendeteksi cahaya tersebut sebagai input jika cahaya tersebut terdeteksi berada di rentang warna ruang HSV yang diatur. *Push button move* dan *push button drag* dikondisikan dengan program arduino supaya ketika *push button move* ditekan maka cahaya LED akan menyala dan memberi *input* ke program *computer vision* di komputer supaya pointer di dekstop berpindah sesuai pada koordinat yang terdeteksi, sedangkan untuk *push button drag* ketika ditekan akan membuat cahaya LED menyala dan memberi *input* ke program *computer vision* di komputer supaya pointer di dekstop mengklik sesuai pada koordinat yang terdeteksi. Program *computer vision* dibuat menggunakan pustaka *opencv* dan penyesuaian koordinat dibuat dengan menggunakan pustaka *pyautogui*. Berikut ini adalah *flowchart* program yang dibutuhkan dalam penelitian ini.



Gambar 9. Blok diagram *interactive whiteboard*





**Gambar 10.** Blok diagram persiapan pengujian tingkat presisi

### 3.5.3 Perancangan Pengujian

Pengujian efektivitas *interactive whiteboard* dilakukan dengan menghitung nilai MAE (*Mean Absolute Error*), MRE (*Mean Relative Error*), dan standar deviasi dari pendeteksian pena LED. Pena LED akan dibiarkan menyala saat kamera merekam dengan variasi resolusi kamera sebesar 320p, 480p, dan 960p. Pengujian pertama dilakukan berdasarkan piksel gambar hasil deteksi pena LED. Kamera akan merekam dalam rentang waktu tertentu hingga didapatkan citra yang berisi cahaya dari pena LED dengan lokasi pointer yang berbeda pada 140 kali pengulangan untuk setiap resolusi kamera yang digunakan. Pada setiap resolusi kamera akan dicoba sebanyak 5 kali pengulangan. Koordinat yang terdeteksi akan dicari tingkat presisinya menggunakan jarak Euclidean (d) 2 dimensi menggunakan rumus berikut,

$$d = \frac{\sum_i^n \sqrt{(x_i - \mu_x)^2 + (y_i - \mu_y)^2}}{n} \quad (2.19)$$

Di mana  $x$  dan  $y$  merupakan koordinat yang terdeteksi,  $\mu$  adalah rata-rata dari koordinat, dan  $n$  adalah jumlah data.

Pengujian kedua dilakukan berdasarkan jarak antara cahaya LED di dinding proyeksi dan pointer desktop di komputer dalam satuan sentimeter menggunakan alat ukur penggaris. Pergantian posisi dilakukan sebanyak 30 kali dengan 3 pergantian kalibrasi untuk resolusi kamera sebesar 320 x 240 piksel, 640 x 480 piksel, dan 960 x 720 piksel. Setelah nilai kesalahan pendeteksian untuk titik dideteksi, kesalahan pendeteksian saat menarik garis pada *Microsoft Paint* dicari dengan menyiapkan 5 garis lurus dengan panjang 30 cm terlebih dulu dengan 6 titik hitung kesalahan. Kemudian dengan menggunakan pena LED dicoba menggambar garis lurus mengikuti garis acuan yang telah dibuat. Kesalahan pendeteksian akan dihitung pada 6 titik hitung tersebut menggunakan alat ukur penggaris.

Dari setiap perbedaan jarak tersebut akan dicari nilai rata-rata untuk setiap resolusi yang digunakan. Efektivitas dari metode pengenalan warna pada LED warna hijau ini akan dibuktikan berdasarkan tingkat akurasi yang dicari melalui persamaan rata-rata dan standar deviasi berikut,

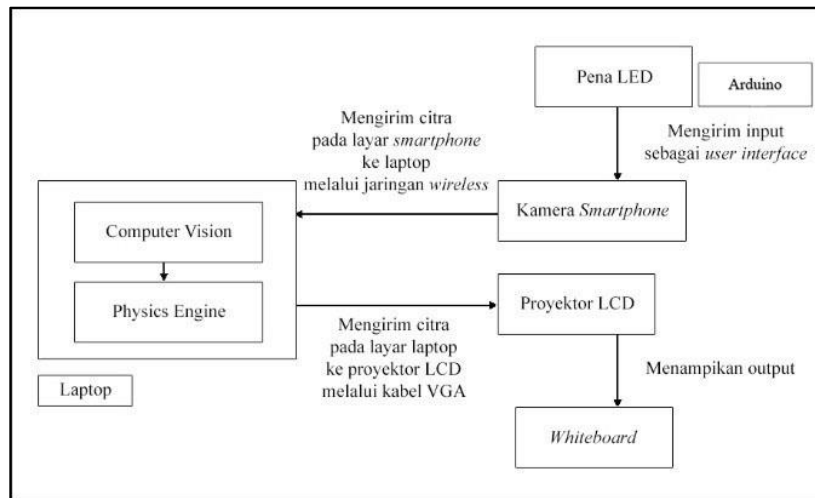
$$\mu_d = \frac{1}{n} \sum_i^n |e| \quad (2.20)$$

$$\sigma_d = \sqrt{\frac{\sum(\bar{e} - e)^2}{n - 1}} \quad (2.21)$$

Di mana  $e$  adalah hasil kesalahan seberapa jauh jarak yang terdeteksi pada pengujian kedua dan  $n$  adalah jumlah cuplikan gambar yang diambil. Selain itu, standar deviasi ( $\sigma$ ) dihitung untuk mengetahui tingkat presisinya. Semakin besar nilai  $\sigma$  maka semakin kecil tingkat presisinya. Nilai *delay* waktu pun dihitung untuk menulis garis.

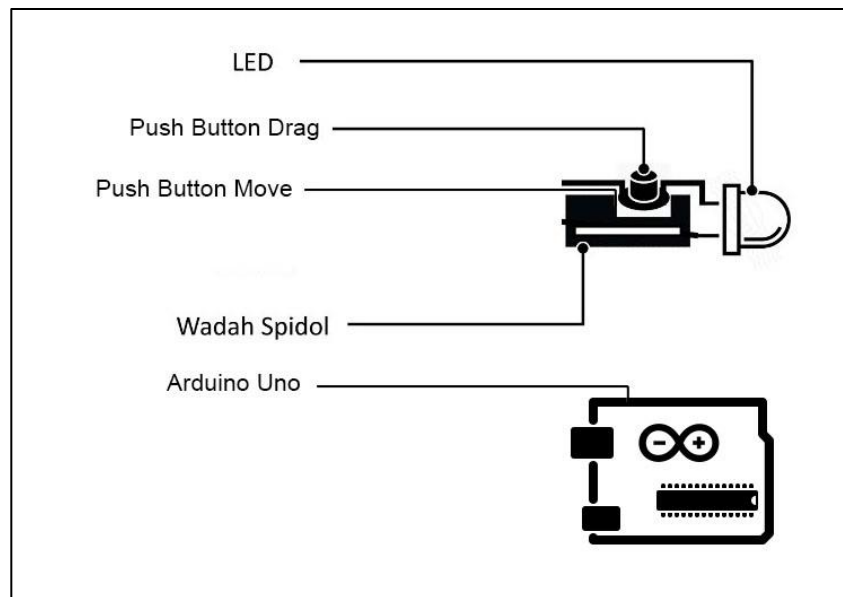
Setelah penghitungan tersebut dilakukan maka *interactive whiteboard* dicoba untuk berinteraksi dengan program *Microsoft Paint* dan *Algoodo*.

### 3.6 Blok Diagram Sistem

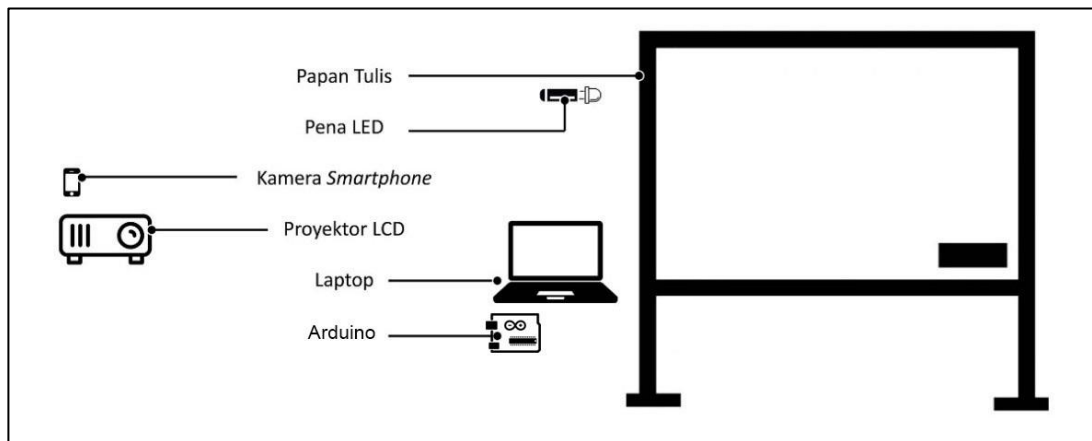


Gambar 11. Blok diagram sistem

### 3.7 Skema Alat

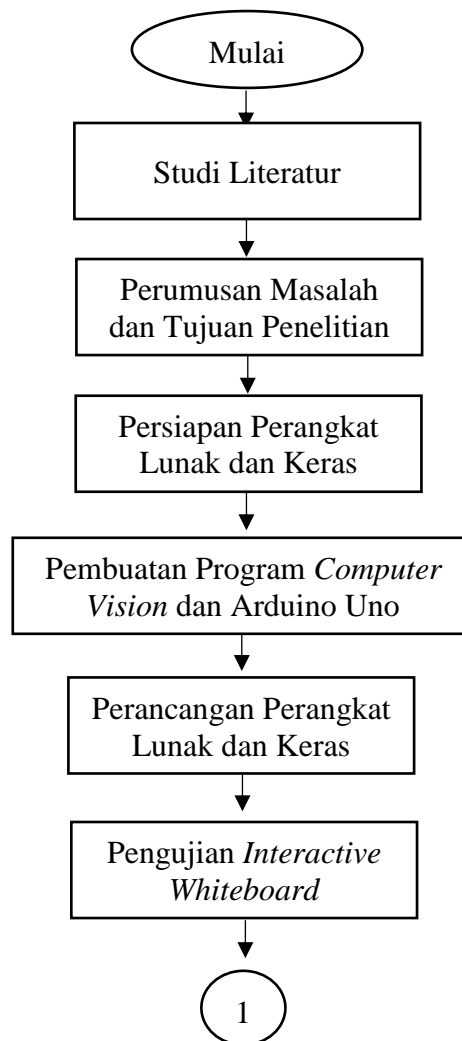


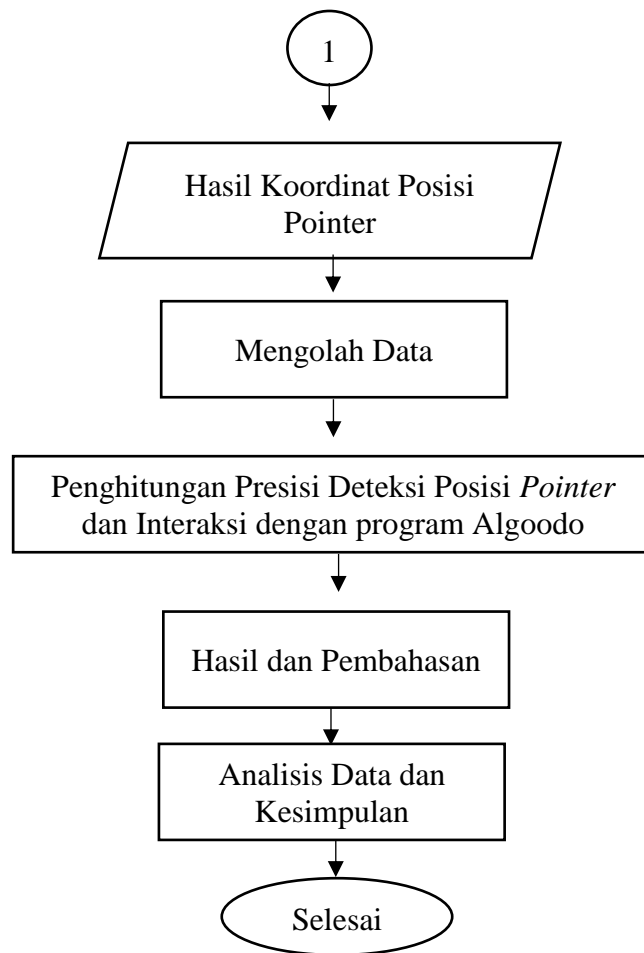
Gambar 12. Rancang bangun pena LED cahaya tampak



**Gambar 13.** Rancang bangun *interactive whiteboard*

### 3.8 Diagram Alir Penelitian





**Gambar 14.** Diagram alir penelitian

## BAB IV HASIL DAN PEMBAHASAN

### 4.1 Pra Eksperimen

#### 4.1.1 Persiapan Pengujian *Interactive Whiteboard*

Pengambilan video dilakukan dengan menggunakan kamera *smartphone* Android Samsung J5 Prime, bertempat di laboratorium instrumentasi fisika UNJ. Pengujian efektivitas terbagi menjadi dua, yaitu: pengujian efektivitas berdasarkan piksel kamera dengan satuan piksel dan pengujian efektivitas berdasarkan layar *interactive whiteboard* dengan satuan sentimeter. Pada setiap pengujian, pertama kali diperlukan *setting* alat serta kalibrasi ukuran proyeksi komputer ke dinding dan ukuran layar *desktop* di komputer. Kemudian untuk pengujian efektivitas pertama pengambilan video ketika menggunakan pena LED *interactive whiteboard* dilakukan sebanyak 3 kali untuk masing-masing resolusi kamera sebesar 360 x 240 piksel, 640 x 480 piksel, dan 960 x 720p. Setelah rekaman diperoleh, rekaman dipindahkan ke komputer dan diproses lebih lanjut dengan algoritma pendeteksi lokasi cahaya LED untuk dicari koordinat jarak Euclidean dari hasil deteksi dalam satuan piksel. Untuk pengujian efektivitas kedua, jarak antara lokasi cahaya LED di dinding proyeksi dan lokasi *pointer* di layar *desktop* dilakukan. Algoritma pengenalan warna dibuat menggunakan karakter ruang warna HSV sebagai berikut,

**Tabel 2.** Karakteristik Ruang Warna HSV untuk *Image Masking*

No	Nama	Rentang nilai
1	<i>Hue</i>	35 – 85
2	<i>Saturation</i>	25 – 130
3	<i>Value</i>	190– 250

Penggunaan *interactive whiteboard* kemudian dicoba pada aplikasi *Microsoft Paint* dan *Algoodo* untuk melihat kualitas penggunaan alat yang dibuat. Hasil percobaan dilihat berupa gambar saat menggunakan aplikasi tersebut.

#### 4.1.2 Setting Alat *Interactive Whiteboard*

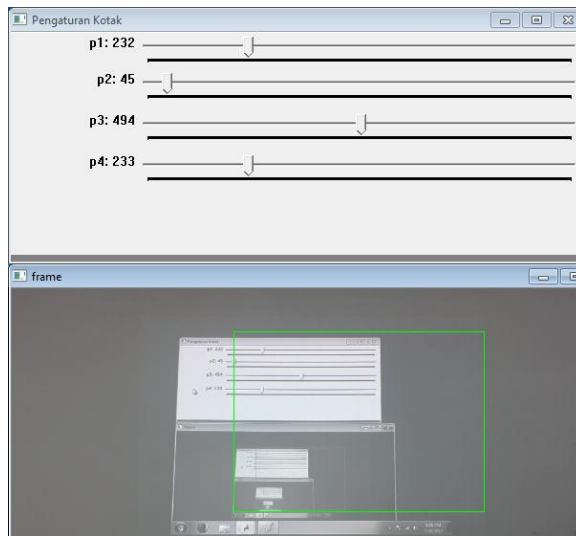
Beberapa *setting* alat diperlukan sebelum menjalankan program, di antaranya adalah menyiapkan proyektor sejauh kurang lebih 1.2 meter, meletakkan kamera *smartphone* di bawah proyektor, dan menghubungkan arduino yang sudah terpasang *pointer* LED ke laptop. Kecerahan proyektor diatur dengan tingkat kecerahan paling rendah dan aplikasi *IP Webcam* digunakan sebagai aplikasi perekam video Hasil *setting* adalah sebagai berikut:



**Gambar 15.** Setting alat *Interactive whiteboard*

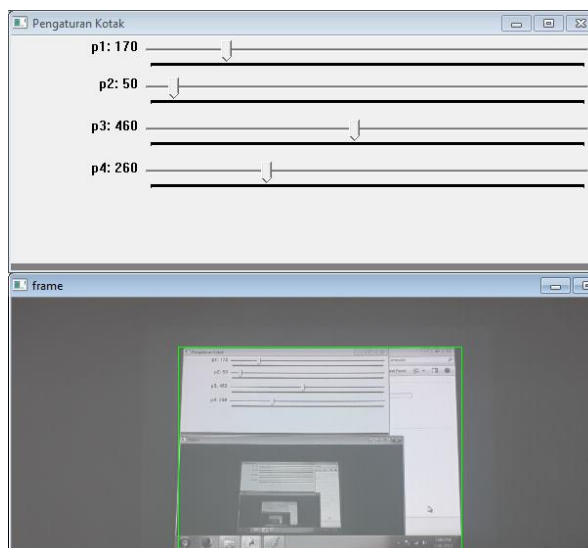
#### 4.1.3 Kalibrasi Layar Proyeksi LCD dengan Layar Desktop Komputer

Kalibrasi dilakukan untuk memetakan proyeksi gambar pada dinding dengan resolusi tertentu dengan layar desktop dengan resolusi tertentu. Hal ini pun dilakukan supaya layar sesuai ukurannya. Pada saat program pertama kali dijalankan, jendela pengaturan kalibrasi muncul.



**Gambar 16.** Persiapan kalibrasi pada layar desktop komputer

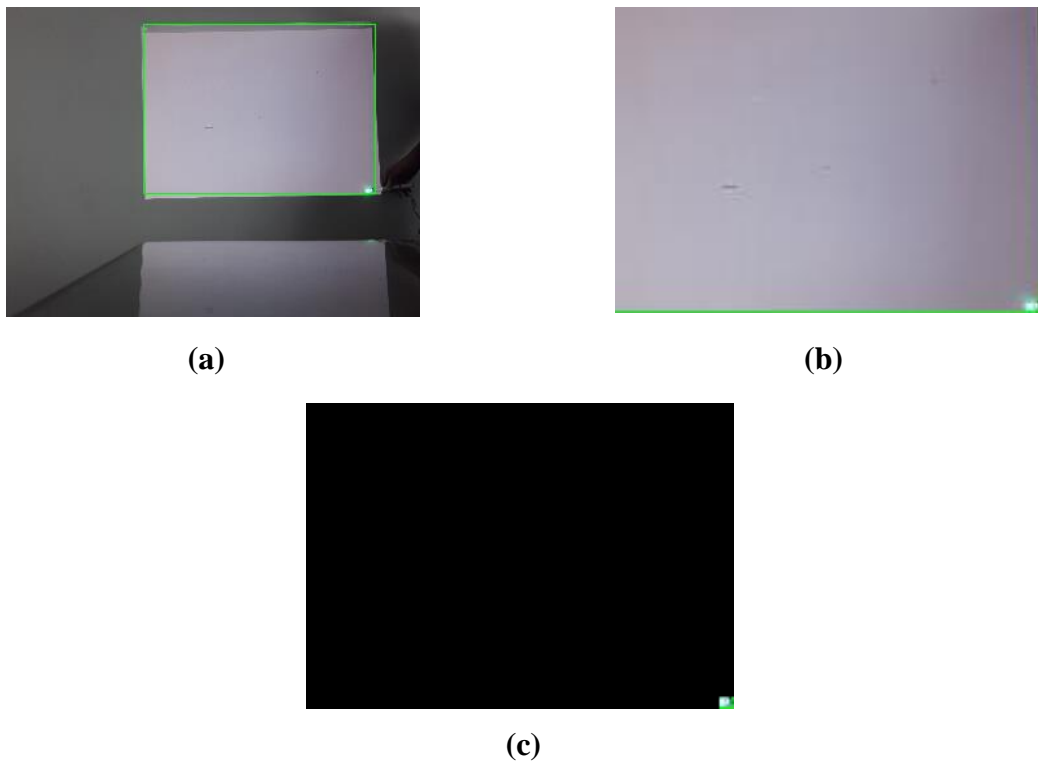
Supaya posisi cahaya LED di dinding proyeksi dapat berada di posisi yang mendekati sama dengan pointer pada layar komputer, maka kalibrasi harus sesuai. Penyesuaian kalibrasi dilakukan dengan menggeser variabel p1, p2, p3, dan p4 pada jendela pengaturan kalibrasi hingga didapat *region of interest cropping* sebagai berikut.



**Gambar 17.** Kalibrasi pada layar desktop komputer telah dilakukan



Setelah kalibrasi dilakukan dengan sesuai maka ketika *interactive whiteboard* digunakan akan mendapatkan hasil *region of interest cropping* dan *masking image region* seperti pada gambar berikut.



**Gambar 18.** Deteksi pena LED *move* pada (a) hasil rekaman kamera, (b) hasil *region of interest cropping*, dan (c) *masking image region of interest* dengan resolusi kamera 320 x 240 di salah satu *frame*

Gambar 19 menunjukkan frame pada suatu video pada waktu yang sama. Pada gambar 19a ditampilkan hasil perekaman video oleh kamera *smatphone* Android dengan resolusi 320 x 320 piksel. Kemudian dari program di komputer, hasil perekaman di-*crop* berdasar jendela kalibrasi seperti yang ditunjukkan gambar 19b. Cahaya LED yang terdeteksi akan terlihat pada gambar 19c.

## 4.2 Presisi Deteksi *Pointer* LED pada Kamera

### 4.2.1 Persiapan Eksperimen

Deteksi LED dilakukan dengan menggunakan metode *Color Detection* berdasar ruang warna HSV. Untuk melakukan pendeteksian tersebut, digunakan bahasa pemrograman Python dan pustaka program OpenCV yang memudahkan dalam mendeteksi objek dengan menggunakan *syntax* “*cv2.inRange()*”. Dengan menggunakan algoritma yang sesuai dengan gambar 10 dan 11, pengambilan video di laboratorium instrumentasi dilakukan menggunakan kamera *smartphone* dengan resolusi sebesar 320 x 240 piksel, 640 x 480 piksel, dan 960 x 720 piksel serta jarak antara dinding dan kamera sejauh 1.2 meter. Deteksi pointer LED tersebut dilakukan menggunakan aplikasi android *IP Webcam* untuk mentransfer video tiap *frame* ke komputer. Kemudian video tiap *frame* tersebut diolah di komputer menggunakan program yang mengoperasikan citra digital dengan metode *Color Detection*. Pustaka yang digunakan pada program tersebut adalah OpenCV dan PyAutoGUI.

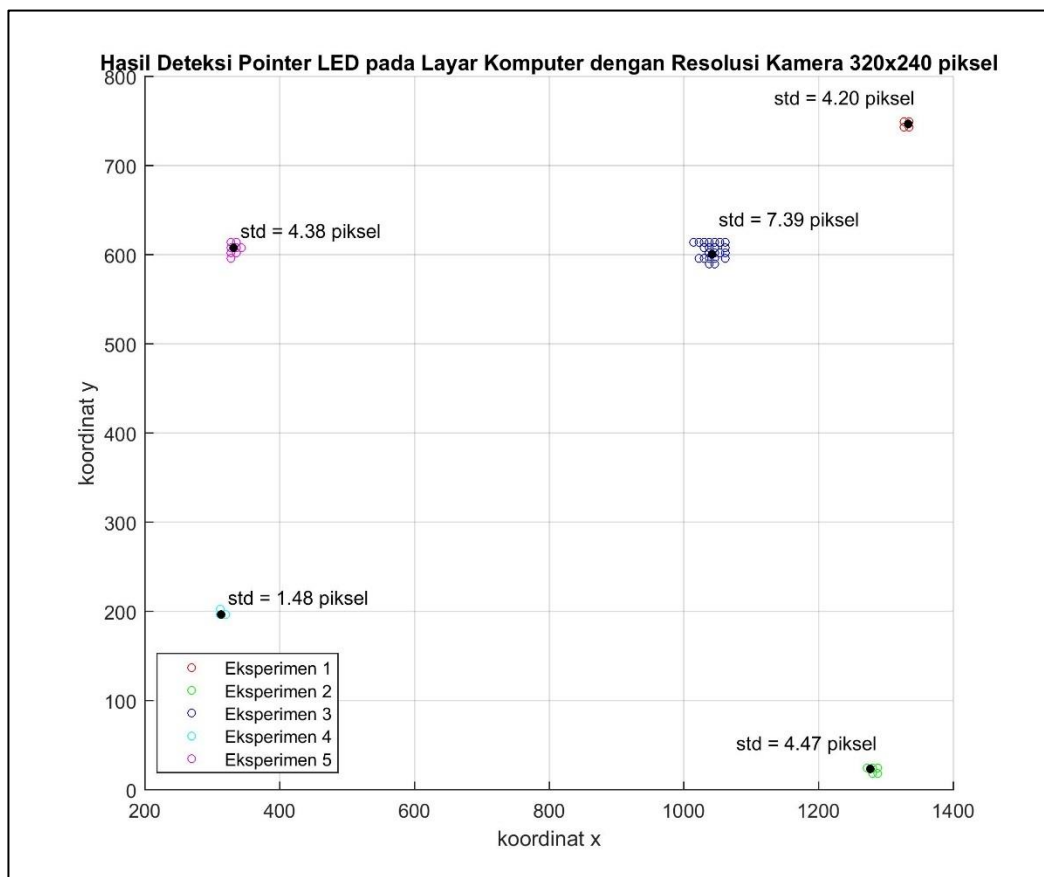
Setelah persiapan dilakukan, algoritma pemrograman yang digunakan dalam penghitungan presisi deteksi lokasi pointer adalah sebagai berikut.

1. Mulai
2. Menulis alamat IP yang tertera di aplikasi *IP webcam* ke program *interactive whiteboard*
3. Melakukan pengulangan sebanyak jumlah frame
  - a. Mengambil sebuah *frame* pada video
  - b. Melakukan kalibrasi *region of interest cropping* pada proyeksi layar *interactive whiteboard*
  - c. Mengubah ruang warna frame dari RGB ke HSV
  - d. Melakukan *image masking* untuk mendeteksi pena LED
  - e. Mendeteksi cahaya paling terang pada sebuah piksel
  - f. Melakukan penghitungan jarak Euclidean pada piksel yang terdeteksi
  - g. Menyimpan data piksel paling terang berupa koordinat X dan koordinat Y ke dalam sebuah *list*
4. Menyimpan data ke dalam file *Microsoft Excel*

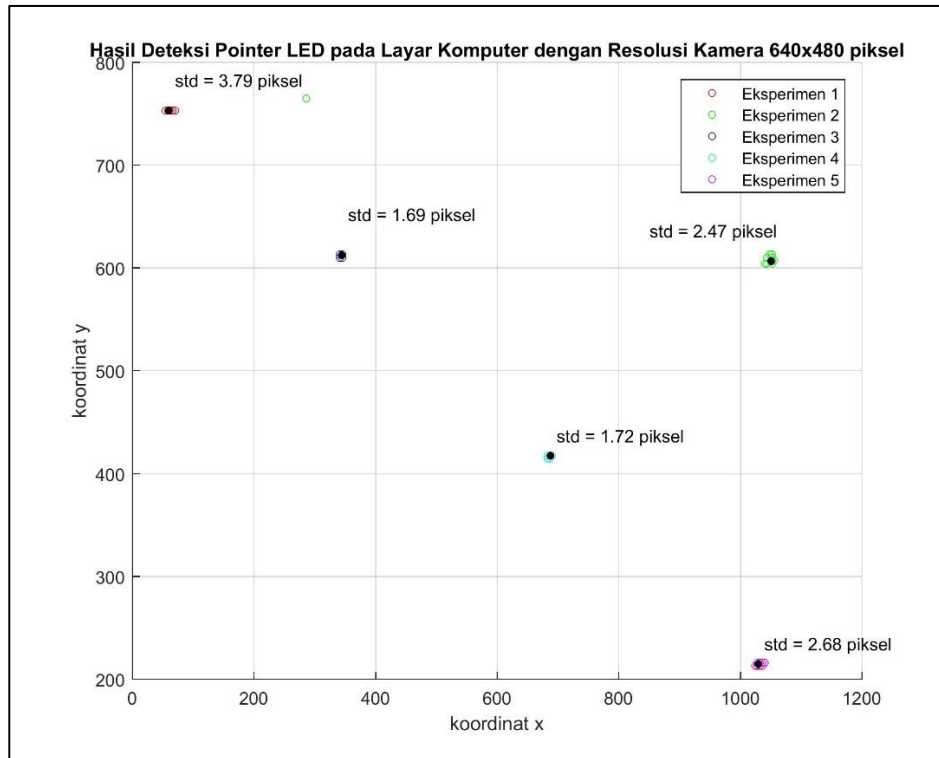
5. Menentukan 5 titik pendeteksian yang dihitung pada 150 frame tiap titik
6. Melakukan pengulangan sebanyak jumlah eksperimen
  - a. Mencari nilai rata-rata dari 150 data nilai koordinat x dan koordinat y
  - b. Menghitung jarak Euclidean antara nilai rata-rata dan tiap 150 data sebagai standar deviasi
  - c. Menghitung rata-rata jarak Euclidean tersebut sebagai standar deviasi
  - d. Membuat plot ke dalam koordinat Kartesius
7. Selesai

#### 4.2.2 Hasil Eksperimen

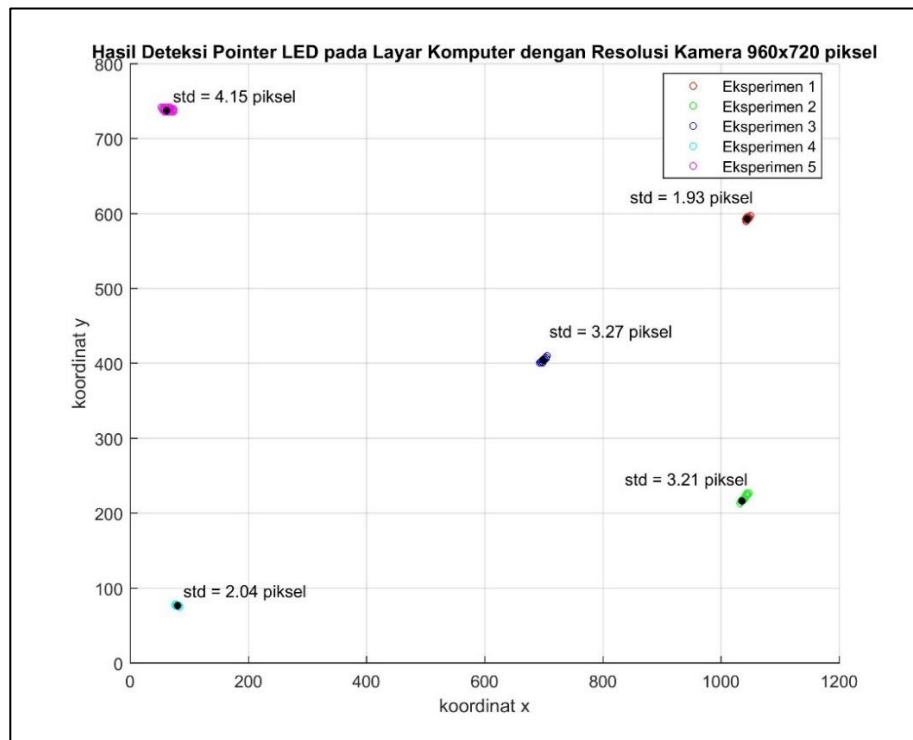
Dari eksperimen untuk menghitung tingkat presisi deteksi *pointer* LED pada kamera dengan resolusi berbeda yang telah dilakukan, didapatkan tiga koordinat kartesius yang berisi hasil deteksi *pointer* LED sebagai berikut.



**Gambar 19.** Hasil Deteksi Pointer pada Layar Komputer dengan Resolusi Kamera 320 x 240 piksel



**Gambar 20.** Hasil Deteksi Pointer pada Layar Komputer dengan Resolusi Kamera 640 x 480 piksel



**Gambar 21** Hasil Deteksi Pointer pada Layar Komputer dengan Resolusi Kamera 640 x 480 piksel

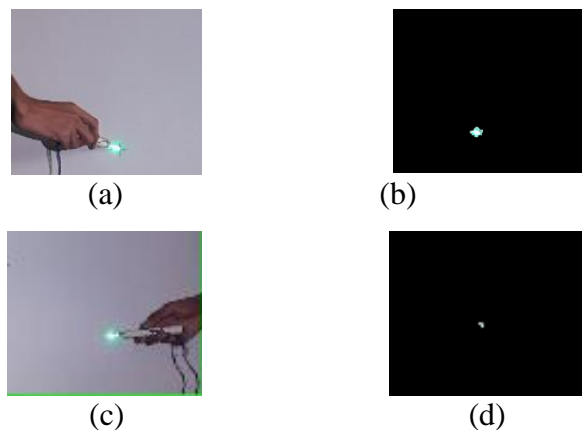
Pada hasil yang didapat menggunakan 3 resolusi kamera yang berbeda seperti yang ditunjukkan gambar 20, gambar 21, dan gambar 22 terlihat bahwa standar deviasi yang diperoleh berkisar antara 1.5 hingga 7.5 di mana nilai ini merepresentasikan tingkat presisi deteksi pointer LED pada kamera. Tingkat presisi tersebut ditunjukkan pada tabel berikut.

**Tabel 3. Tingkat Presisi Deteksi Pointer LED pada Kamera**

No	Standar Deviasi (piksel)		
	Resolusi Kamera 320 x 240 piksel	Resolusi Kamera 640 x 480 piksel	Resolusi Kamera 960 x 720 piksel
1	4.38	3.79	4.15
2	1.48	1.69	1.93
3	7.39	1.72	3.27
4	4.2	2.47	3.21
5	4.47	2.68	2.04

Berdasar tabel di atas, dapat dilihat bahwa nilai rata-rata presisi deteksi pointer LED menggunakan resolusi kamera 320 x 240, 640 x 480, dan 960 x 720 piksel berturut-turut adalah 4.38, 2.47, dan 2.92 piksel. Menggunakan resolusi kamera 320 x 240 piksel memiliki tingkat presisi yang lebih rendah dibanding dengan resolusi kamera 640 x 480 dan 960 x 720 piksel.

Sebagai perbandingan antara tingkat presisi yang rendah dan tinggi, yaotu 7.39 dan 1.72 adalah sebagai berikut.



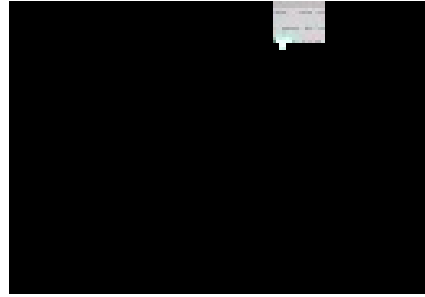
**Gambar 22.** Hasil Deteksi Pointer LED pada eksperimen ke-3 pada (a) (b) resolusi kamera 640 x 480 piksel dan pada (c) (d) resolusi kamera 320 x 240

Berdasar gambar 23a dan 23b terlihat bahwa ketika menggunakan resolusi kamera 640 x 480 piksel dengan tingkat presisi 1.72 mendapatkan hasil deteksi dengan luas deteksi yang lebih besar dibanding dengan gambar 23c dan 23d ketika menggunakan resolusi kamera 320 x 240 piksel dengan tingkat presisi 7.39. Dengan kata lain, resolusi kamera yang rendah akan mengurangi tingkat presisinya karena luas deteksi yang dihasilkan kecil.

Dalam penggunaannya, *interactive whiteboard* ini tidak dapat dengan baik menangani pengenalan pointer LED ketika cahaya LED terlalu menyebar pada layar proyeksi LCD seperti yang ditunjukkan gambar berikut.



(a)



(b)

**Gambar 23.** Deteksi pena LED pada (a) hasil region of interest cropping kamera, dan (b) masking image region of interest dengan resolusi kamera 320 x 240 di salah satu *frame*

Pada gambar di atas terdapat cahaya dalam rentang ruang warna HSV yang diatur ikut terdeteksi. Hal ini dapat membuat pointer pada layar komputer bergeser-geser tak beraturan dengan simpangan tertentu karena hasil deteksi tersebut seperti yang ditunjukkan gambar 20 pada eksperimen ke-3. Selain itu, intensitas cahaya yang terlalu tinggi, salah satunya dari matahari, seperti ketika berada pada jarak yang dekat dengan jendela pada siang hari pun dapat membuat hasil deteksi mengalami gangguan seperti membaca lokasi pointer LED yang tidak seharusnya. .

### 4.3 Presisi Deteksi Pointer LED pada Layar Proyeksi LCD

#### 4.3.1 Persiapan Eksperimen

Persiapan awal penghitungan tingkat akurasi deteksi pointer pada layar proyeksi LCD ini hampir sama dengan persiapan awal yang dilakukan pada presisi penghitungan deteksi pointer pada kamera. Perbedaan yang ada pada persiapan eksperimen ini adalah algoritma menghitung nilai standar deviasi untuk mengetahui tingkat presisinya. Ada pun algoritma pemrograman yang digunakan dalam penghitungan akurasi dan presisi pendeteksian cahaya LED adalah sebagai berikut.

1. Mulai
2. Rekam Video
3. Melakukan pengulangan sebanyak 30 kali dalam posisi yang berbeda
  - a. Mengambil sebuah *frame* pada video

- b. Melakukan kalibrasi *region of interest cropping* pada proyeksi layar *interactive whiteboard*
  - c. Mengubah ruang warna *frame* dari RGB ke HSV
  - d. Melakukan *image masking* untuk mendeteksi pena LED
  - e. Mendeteksi cahaya paling terang pada sebuah piksel
  - f. Mengarahkan cahaya LED ke titik acuan
  - g. Mengukur jarak kesalahan antara titik acuan dengan hasil pendeteksian menggunakan penggaris
  - h. Mencatat data ke dalam file *Microsoft Excel*
4. Menghitung standar deviasi sebagai nilai dari presisi
  5. Selesai

Algoritma tersebut dilakukan menggunakan variasi resolusi kamera.

#### 4.3.2 Hasil Eksperimen

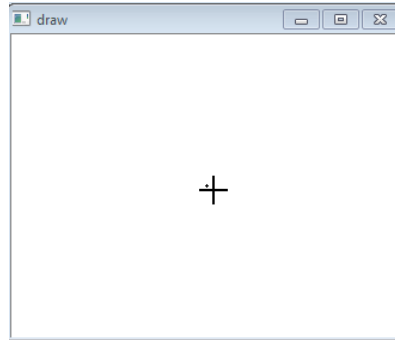
Tabel di atas adalah nilai kesalahan rata-rata dan standar deviasi dari 10 data untuk tiap eksperimen dan resolusi kamera.

**Tabel 4.** Data hasil nilai kesalahan deteksi lokasi Pena LED

Eksperimen	Resolusi Kamera 320 x 240 piksel		Resolusi Kamera 640 x 480 piksel		Resolusi Kamera 960 x 720 piksel	
	$\mu_a$ (cm)	$\sigma_a$ (cm)	$\mu_a$ (cm)	$\sigma_a$ (cm)	$\mu_a$ (cm)	$\sigma_a$ (cm)
1	1.62	0.84	0.87	0.43	0.72	0.32
2	2.07	0.79	1.01	0.35	0.97	0.25
3	2.07	0.83	1.09	0.48	0.77	0.33

Setiap eksperimen dari pengukuran jarak antara cahaya LED yang diposisikan tepat berada di titik seperti pada gambar di bawah. Kemudian hasil deteksi lokasi pointer di layar proyeksi LCD ditunjukkan simbol positif seperti gambar berikut,





**Gambar 24.** Perbedaan jarak antara lokasi pena LED ditunjukkan simbol titik dan hasil deteksi ditunjukkan simbol positif

Dengan menghitung nilai rata-rata dan standar deviasi dari tabel 5, didapat kesalahan absolut rata-rata untuk pointer LED sebagai berikut,

*Tabel 5.* Hasil nilai kesalahan absolut pendeteksian lokasi pena LED

Resolusi Kamera (px)	240p	480p	720p
Error Deteksi (cm)	$1.92 \pm 0.82$	$0.99 \pm 0.42$	$0.82 \pm 0.3$

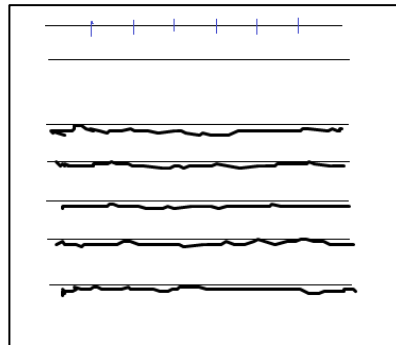
Selain pena LED yang diam, akurasi membuat garis lurus dengan pena LED pun dihitung dan menghasilkan tabel sebagai berikut,

**Tabel 6.** Nilai kesalahan pendeteksian menulis garis lurus menggunakan pena LED

Eksperimen	<i>error</i> (cm)
1	0.7
2	0.4
3	0.57
4	0.37
5	0.45

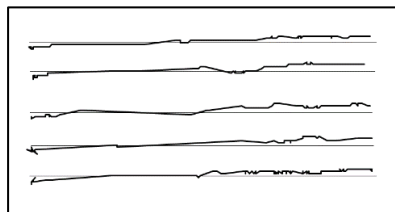
Tabel di atas berasal dari *error* pendeteksian rata-rata dari setiap eksperimen antara garis lurus acuan dengan garis yang ditulis dengan pena LED sebanyak 5 garis lurus dengan panjang 30 cm yang dihitung pada 6 titik. Dari tabel 5 didapat nilai kesalahan menulis garis lurus sebesar  $0.50 \pm 0.14$  cm. Dengan kata lain,

akurasi bernilai 0.5 cm dan presisinya bernilai 0.14 cm. Gambar garis lurus menggunakan pena LED adalah seperti yang ditunjukkan gambar berikut,

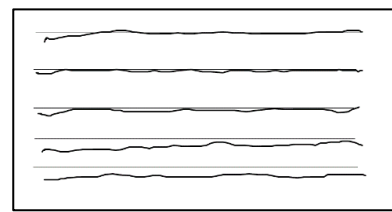


**Gambar 25.** Hasil menulis 5 garis lurus dengan menggunakan pena LED

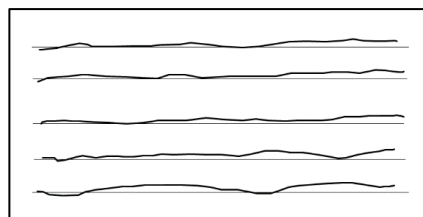
Saat menulis garis, hasil deteksi memiliki *delay* waktu untuk mengikuti *pointer* LED. Penulisan garis telah diuji menghasilkan gambar dan tabel berikut,



(a)



(b)



(c)

**Gambar 26.** Hasil menulis garis lurus untuk menguji *delay* waktu pada (a) resolusi kamera 320 x 240 piksel (b) resolusi kamera 640 x 480 piksel, dan (c) resolusi kamera 960 x 720 piksel

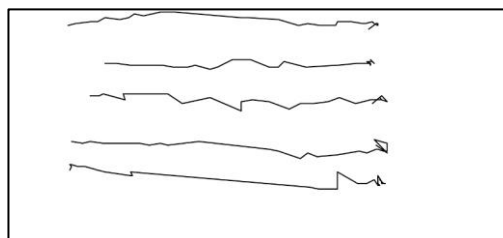
**Tabel 7.** Waktu yang diperlukan pointer LED untuk menulis garis lurus

No	Waktu (s)		
	Resolusi Kamera 320 x 240 piksel	Resolusi Kamera 640 x 480 piksel	Resolusi Kamera 960 x 720 piksel
1	17.92	8.25	7.61
2	17.34	8.35	6.99
3	15.36	8.08	7.28
4	15.46	8.6	8.07
5	19.71	8.19	6.75

Dari hasil gambar 27 dan tabel 7 dapat dilihat waktu yang dibutuhkan pointer LED untuk menulis garis lurus dengan panjang yang sama pada *interactive whiteboard* menggunakan resolusi kamera 320 x 240, 640 x 480, dan 960 x 720 piksel berturut-turut adalah 17.16, 8.29, dan 7.34 detik. Menggunakan resolusi kamera 320 x 240 piksel memerlukan waktu yang lebih lama dibanding dengan resolusi kamera 640 x 480 dan 960 x 720 piksel.

#### 4.4 Penggunaan Kamera yang Berbeda

Percobaan telah diuji dengan kamera dari *smartphone* yang berbeda, yaitu Samsung J5 Prime, Samsung Galaxy S Duos, dan LG X Power. Pada hasil yang didapat terlihat di intensitas cahaya yang tinggi pada siang hari dekat dengan jendela hasilnya sebagai berikut untuk Samsung J5 Prime:

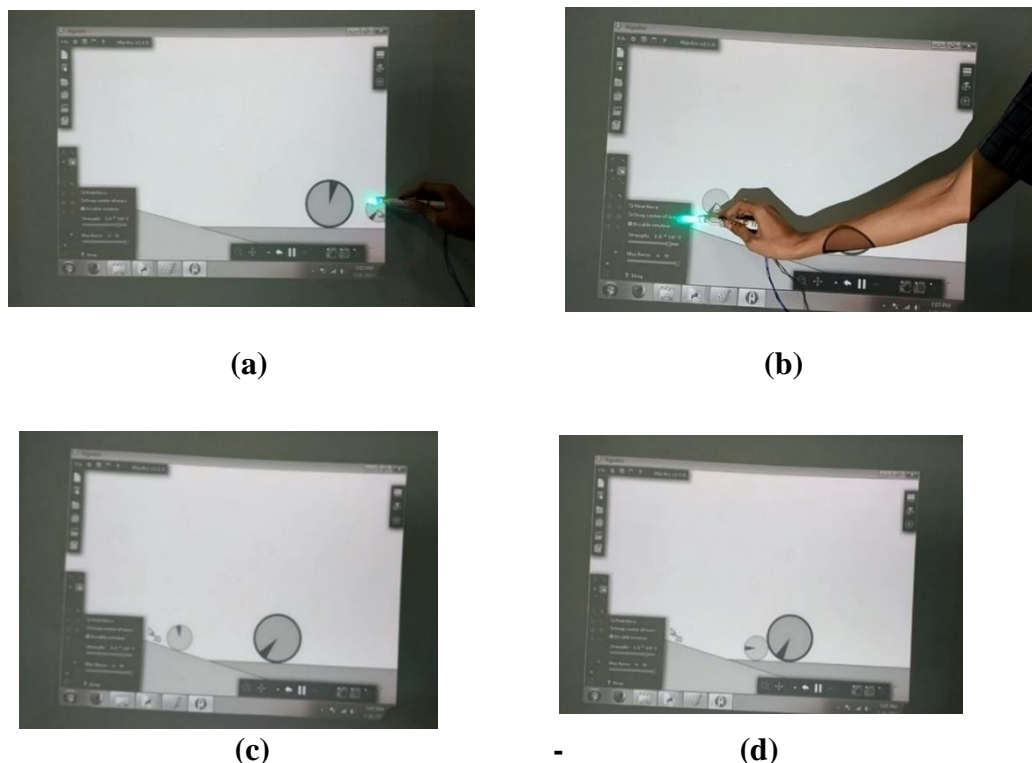


**Gambar 27.** Hasil Menulis di Microsoft Paint Menggunakan Kamera Samsung J5 Prime

Namun untuk kamera *smartphone* Samsung Galaxy S Duos dan LG X Power, pointer LED tidak dapat terdeteksi dengan baik. Hal ini dimungkinkan terjadi karena hasil perekaman menggunakan kamera *smartphone* menggunakan aplikasi android *IP Webcam* memiliki sensitivitas yang kurang tinggi untuk mendeteksi cahaya LED dengan tingkat saturasi yang rendah.

#### 4.5 Interaksi dengan Aplikasi Simulasi Fisika

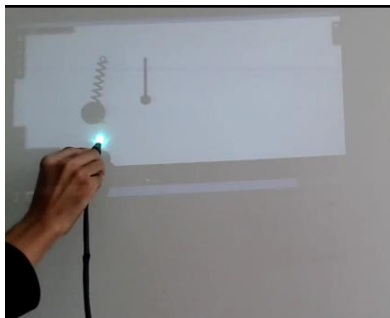
Simulasi program fisika Algoodo dijalankan menggunakan pena LED. Dalam penelitian ini terdapat 3 simulasi fisika yang diujikan, yaitu: gerak menggelinding dan tumbukan, gerak benda yang terikat pada pegas, serta gerak benda yang terikat pada seutas tali. Pengujian pertama dilakukan untuk simulasi gerak menggelinding dan tumbukan menunjukkan hasil seperti pada gambar di bawah.



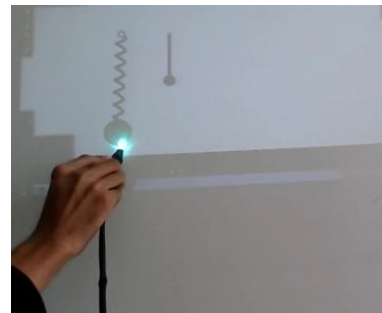
**Gambar 28.** Program simulasi fisika Algoodo dijalankan dengan pena LED *interactive whiteboard*. (a) Lingkaran kecil diklik, (b) dipindahkan, (c) kemudian dilepas, hingga (d) menggelinding dan menumbuk lingkaran besar.

Dua lingkaran dan satu bidang miring pertama dibuat seperti pada gambar 30a. Dua lingkaran tersebut dibuat dengan ukuran yang berbeda di mana satu lingkaran berukuran besar dan lingkaran lainnya berukuran kecil. Lingkaran kecil diklik untuk dipindahkan menggunakan pena LED. Pena LED dinyalakan dan digeser hingga seperti yang ditunjukkan gambar 30b kemudian untuk melepas lingkaran kecil pena LED dimatikan seperti yang ditunjukkan gambar 30c. Lingkaran kecil tersebut akan menggelinding dan menumbuk lingkaran besar seperti yang ditunjukkan gambar 30d.

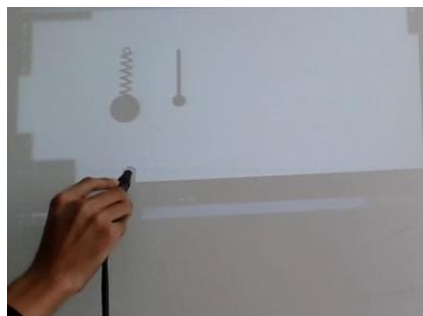
Pengujian kedua dan ketiga untuk gerak benda yang terikat pada pegas dan gerak benda yang terikat pada seutas tali dilakukan pada satu tempat yang sama pada aplikasi Algoodo. Pengujian kedua ditunjukkan seperti pada gambar di bawah.



(a)



(b)

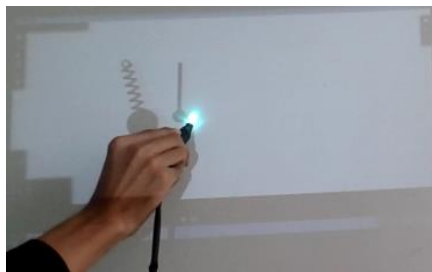


(c)

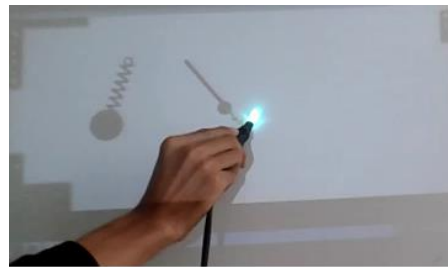
**Gambar 29.** Program simulasi fisika Algoodo dijalankan dengan pena LED *interactive whiteboard*. (a) Lingkaran diklik, (b) ditarik ke bawah (c) kemudian dilepas hingga pegas bekerja dan lingkaran bergerak naik turun.

Lingkaran dan sebuah pegas dibuat dan dihubungkan seperti pada gambar 30a. Lingkaran diklik untuk ditarik menggunakan pena LED. Pena LED dinyalakan dan digeser hingga seperti yang ditunjukkan gambar 31b kemudian untuk melepas lingkaran pena LED dimatikan seperti yang ditunjukkan gambar 31c. Lingkaran tersebut bergerak ke atas dan ke bawah. Namun, penarikan lingkaran harus ditarik ke bawah tanpa ada pergeseran ke arah kiri atau kanan supaya pegas hanya bekerja ke atas dan ke bawah.

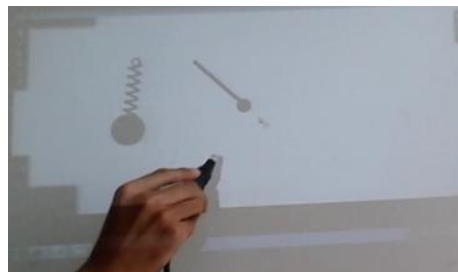
Pengujian ketiga ditunjukkan seperti pada gambar di bawah.



(a)



(b)



(c)

**Gambar 30.** Program simulasi fisika Algoodo dijalankan dengan pena LED *interactive whiteboard*. (a) Lingkaran diklik, (b) ditarik ke kanan (c) kemudian dilepas hingga lingkaran berayun ke kanan dan kiri

Lingkaran dan sebuah tali dibuat dan dihubungkan seperti pada gambar 30a. Lingkaran diklik untuk ditarik menggunakan pena LED. Pena LED dinyalakan dan digeser hingga seperti yang ditunjukkan gambar 31b kemudian untuk melepas lingkaran pena LED dimatikan seperti yang ditunjukkan gambar 31c. Lingkaran tersebut bergerak ke kanan dan kiri.

Hasil di atas menunjukkan *interactive whiteboard* dapat dijalankan untuk program simulasi fisika Algoodo meski dalam penggunaannya dapat terjadi *delay* sehingga efektivitasnya kurang memadai.

## BAB V

### KESIMPULAN

#### 5.1 Kesimpulan

Dari penelitian dan evaluasi yang telah dilakukan, dapat disimpulkan bahwa:

1. *Interactive whiteboard* menggunakan pena LED cahaya tampak berbasis kamera *smartphone* dapat dibuat dan digunakan dengan jarak antara proyektor LCD dan dinding  $\pm 1.2$  meter dan kamera *smartphone* yang ter-*install* aplikasi IP Webcam dengan kualitas kamera yang memadai. Selain itu, proyektor perlu diatur dengan tingkat kecerahan yang rendah.
2. Tingkat presisi dengan resolusi kamera 640 x 480 dan 960 x 720 piksel lebih tinggi dengan standar deviasi 2.47, dan 2.92 piksel dibanding dengan resolusi kamera 320 x 240 piksel. Hal ini dapat disebabkan karena pada resolusi kamera 320 x 240 piksel luas hasil deteksi lebih kecil.
3. *Interactive whiteboard* lebih responsif pada resolusi kamera 640 x 480 dan 960 x 720 piksel dengan waktu 8.29 dan 7.34 detik untuk menulis garis lurus sepanjang 75 cm dibanding dengan resolusi kamera 320 x 240 piksel.
4. Cahaya dari LED akan dapat lebih menyebar pada layar proyeksi LCD sehingga membuat hasil deteksi akan bergeser-geser jika cara memegang *pointer* LED kurang sesuai seperti yang ditunjukkan gambar 23.
5. *Interactive whiteboard* yang dibuat telah diujikan pada aplikasi simulasi fisika dan dapat berjalan cukup baik dengan intensitas. *Pointer* LED dapat memindahkan objek lingkaran dan membuatnya menggelinding, menarik objek lingkaran pada pegas, serta menarik objek lingkaran pada tali.



## 5.2 Saran

Adapun saran-saran yang dapat dipertimbangkan untuk penelitian lebih lanjut adalah sebagai berikut:

1. Dalam proses mengenali warna menggunakan metode *color detection*, ada baiknya menggunakan pemilihan nilai dari HSV menggunakan *artificial neural network* untuk mempelajari warna-warna yang perlu dideteksi dan tidak.
2. Selain menggunakan metode *color detection* sebagai metode untuk mencari lokasi pena LED, ada baiknya dicoba menggunakan metode *object detection* yang lain atau mengembangkan algoritma yang telah dibuat dalam penelitian ini supaya tingkat presisi pengenalan cahaya LED lebih baik.
3. Sebaiknya aplikasi perekaman video pada kamera *smartphone* diperhitungkan pada sisi transfer data melalui *local area network* dengan meninjau *video codec* supaya *delay* waktu dapat diatasi.
4. Sebaiknya dibangun aplikasi perekaman video untuk Android dengan jenis pengkompresan data *loseless compression* supaya kualitas video tidak berkurang dan tetap memiliki sensitivitas yang baik dalam mengenali pena LED serta ukuran file *video* yang di-*transfer* kecil.
5. Sebaiknya dibuatkan *graphic-user interface* dalam penggunaannya supaya lebih praktis.
6. Sebaiknya menggunakan LED yang memiliki tingkat saturasi cahaya lebih tinggi.

## DAFTAR PUSTAKA

- [1] Darmawan, F., Wignjoebroto, S., Sudiarno, A. 2010. *Interactive Whiteboard Menggunakan Aplikasi Wii Remote dengan Pendekatan High Touch Design Process*. Jurusan Teknik Industri, Institut Teknologi Sepuluh Nopember (ITS) Surabaya, Kampus ITS Sukolilo Surabaya 60111.
- [2] Wellner, P., 1991. *The DigitalDesk Calculator: Tangible Manipulation On A Desktop Display*. dalam Prosiding ACM UIST
- [3] Lee, J. C. 2008. *Hacking The Nintendo Wii Remote*. Carnegie Mellon University.
- [4] Yusdianto, A., Sukmaaji, A., Sutanto, T. 2012. *Rancang Bangun Aplikasi Interactive Whiteboard untuk Mendukung Pembelajaran Menggunakan Game Controlling*. S1/Jurusan Sistem Informasi, Sekolah Tinggi Manajemen Informatika & Teknik Komputer Surabaya.
- [5] Soares, C., 2012. *LoCoBoard: Low-Cost Interactive Whiteboard Using Computer Vision Algorithms*. University Fernando PessoaPorto. Portugal.
- [6] Zhang, S., dkk. 2012. *Low-Cost Interactive Whiteboard Using the Kinect*. Department of Software Engineering, Northeast Normal University, Changchun, China.
- [7] Podmajersky, J., 2013. *Interactive Whiteboard*. Slovak University of Technology in Bratislava. Faculty of Informatics and Information Technologies. Jan.
- [8] Liou, W., dan Chang C. 2014. *Utilizing a Low-Cost, Laser-Driven Interactive System (LaDIS) to Improve Learning in Developing Rural Regions*. Educational Technology & Society, 17 (3), 93-107.
- [9] Gonzalez, R.C. dan Woods, R.E., 2007, *Digital Image Processing*, 3rd edition, Prentice Hall.
- [10] Tekalp A. M., 2015. *Digital Video Processing Second Edition*. Massachusetts: Pearson Education, Inc.
- [11] Shapiro, L.G. dan Stockman, G.C., 2001, *Computer Vision*, 1st edition, Prentice Hall.

- [12] Parker, J.R. 1997. *Algorithms for Image Processing and Computer Vision*. John Wiley & Son, Inc. NY,USA.
- [13] Szeliski, R., 2010. *Computer Vision: Algorithms and Applications*, New York: Springer.
- [14] Klette, R. 2014. *Concise Computer Vision: An Introduction into Theory and Algorithms*. London: Springer.
- [15] Plataniotis KN, Venetsanopoulos AN. 2000. *Color Image Processing and Applications*. New York: Springer-Verlag.
- [16] Bradski, G. dan Kaehler, A., 2008, *Learning OpenCV: Computer Vision with the OpenCV Library*, O'Reilly Media, Inc., California.
- [17] Forsyth, David A. dan Ponce, Jean. 2003. *Computer Vision A Modern Approach*, international edition. Prentice Hall. USA.
- [18] Sweigart, A. 2017. *PyAutoGUI Documentation*.
- [19] Parangin-angin, B., 2013. Bab II Landasan Teori, Skripsi, Universitas Sumatera Utara.

## LAMPIRAN

### Lampiran 1. Program yang digunakan dalam penelitian

#### Lampiran 1.1 Program *interactive whiteboard* pada Python

```
import numpy as np, cv2, pyautogui, urllib, pyautogui, progressbar, time
from matplotlib import pyplot as plt
from collections import deque
import serial
import time
import imutils

stream=urllib.urlopen('http://192.168.43.1:8080/video')
bytes=""
ser = serial.Serial('COM3', 9600)
buffer_string = ""
last_received = ""
wx = 0
#pass
def nothing(x):
    pass

titlest = 'Pengaturan Kotak'
cv2.namedWindow(titlest)
cv2.resizeWindow(titlest, 600,240)
cv2.createTrackbar('p1',titlest,225,980,nothing)
cv2.createTrackbar('p2',titlest,10,980,nothing)
cv2.createTrackbar('p3',titlest,500,980,nothing)
cv2.createTrackbar('p4',titlest,160,980,nothing)

#mengambil nilai HSV
def getHSV():
    sl = 25
    sh = 130
    vl = 190
    vh = 250
```

```

sl = sl + 10
if vl > 250:
    vl = 190
hl, hh = 35, 85

return sl, sh, hl, hh, vl, vh

```

**#mencari piksel paling terang pada citra**

```

def most_bright(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    radius = 5
    gray = cv2.GaussianBlur(gray, (radius, radius), 0)
    (minVal, maxVal, minLoc, maxLoc) = cv2.minMaxLoc(gray)
    return minVal, maxVal, minLoc, maxLoc

```

**#memfilter citra berdasar ruang warna HSV**

```

def masking_image(frame, hsv, sl, sh, hl, hh, vl, vh):
    mask = cv2.inRange(hsv, (hl, sl, vl), (hh, sh, vh))
    mask = cv2.dilate(mask, None, iterations=16)
    mask = cv2.erode(mask, None, iterations=18)

    output = cv2.bitwise_and(frame, frame, mask = mask)
    return output

```

```

flag = 9;

```

```

while(True):

```

**#mendapatkan frame dari kamera video**

```

bytes+=stream.read(16384)
a = bytes.find('\xff\xd8')
b = bytes.find('\xff\xd9')
if a!=-1 and b!=-1:
    jpg = bytes[a:b+2]
    bytes= bytes[b+2:]
    frame = cv2.imdecode(np.fromstring(jpg, dtype=np.uint8),cv2.IMREAD_COLOR)

```

#### #region of interest cropping

```
p1 = cv2.getTrackbarPos('p1',titlest)
p2 = cv2.getTrackbarPos('p2',titlest)
p3 = cv2.getTrackbarPos('p3',titlest)
p4 = cv2.getTrackbarPos('p4',titlest)
cv2.rectangle(frame,(p1,p2),(p3,p4),(0,255,0),1)
frameROI = frame[p2:p4,p1:p3]
```

#### #mencari nilai faktor kalibrasi layar ROI dan layar dekstop

```
height_video = len(frameROI)
width_video = len(np.rot90(frameROI))
height_desktop = pyautogui.size()[1]
width_desktop = pyautogui.size()[0]
factor_y_2 = (height_desktop / float(height_video))
factor_x_2 = (width_desktop / float(width_video))
```

```
black = np.zeros((height_video,width_video,3), np.uint8)
black[:,:] = (255,255,255)
```

#### #membaca output string dari push button

```
buffer_string = buffer_string + ser.read(ser.inWaiting())
if '\r\n' in buffer_string:
    lines = buffer_string.split('\r\n')
    last_received = lines[-2]
    buffer_string = lines[-1]
```

#### #masking image

```
hsvROI = cv2.cvtColor(frameROI, cv2.COLOR_BGR2HSV)
sl, sh, hl, hh, vl, vh = getHSV()
outputROI = masking_image(frameROI, hsvROI, sl, sh, hl, hh, vl, vh)
```

#### #mencari piksel paling terang pada citra

```
minVal, maxVal, minLoc, maxLoc = most_bright(outputROI)
```

**#menyesuaikan piksel yang terpilih dengan nilai faktor kalibrasi**

```
xv = maxLoc[0]
yv = maxLoc[1]
aaa = xv * factor_x_2
bbb = yv * factor_y_2
y_pos = [aaa,bbb]
```

**## print last\_received**

```
if (aaa != 0 and bbb != 0):
```

```
    #jika push button 1 move ditekan
```

```
    #jika push button 3 click ditekan
```

```
    if (last_received == "3"):
```

```
        wx = wx + 1;
```

```
        if wx == 5:
```

```
            pyautogui.doubleClick(x=y_pos[0], y=y_pos[1])
```

```
            wx = 0
```

```
    #jika push button 1 move ditekan
```

```
    if (last_received == "1"):
```

```
        pyautogui.moveTo(y_pos[0],y_pos[1])
```

```
    #jika push button 2 drag ditekan
```

```
    if (last_received == "2"):
```

```
        pyautogui.mouseDown()
```

```
        flag = 1;
```

```
    #selama button 2 drag ditekan
```

```
    while(True):
```

```
        #mendapatkan frame dari kamera video
```

```
        bytes+=stream.read(16384)
```

```
        a = bytes.find('\xff\xd8')
```

```
        b = bytes.find('\xff\xd9')
```

```
        if a!=-1 and b!=-1:
```

```
            jpg = bytes[a:b+2]
```

```
            bytes= bytes[b+2:]
```

```
            frame = cv2.imdecode(np.fromstring(jpg, dtype=np.uint8),cv2.IMREAD_COLOR)
```

### #region of interest cropping

```
p1 = cv2.getTrackbarPos('p1',titlest)
p2 = cv2.getTrackbarPos('p2',titlest)
p3 = cv2.getTrackbarPos('p3',titlest)
p4 = cv2.getTrackbarPos('p4',titlest)
cv2.rectangle(frame,(p1,p2),(p3,p4),(0,255,0),1)
frameROI = frame[p2:p4,p1:p3]
```

### #mencari nilai faktor kalibrasi layar ROI dan layar dekstop

```
height_video = len(frameROI)
width_video = len(np.rot90(frameROI))
height_desktop = pyautogui.size()[1]
width_desktop = pyautogui.size()[0]
factor_y_2 = (height_desktop / float(height_video))
factor_x_2 = (width_desktop / float(width_video))
```

### #membaca output string dari push button

```
buffer_string = buffer_string + ser.read(ser.inWaiting())
if '\r\n' in buffer_string:
    lines = buffer_string.split('\r\n')
    last_received = lines[-2]
    buffer_string = lines[-1]
```

### #masking image

```
hsvROI = cv2.cvtColor(frameROI, cv2.COLOR_BGR2HSV)
sl, sh, hl, hh, vl, vh = getHSV()
outputROI = masking_image(frameROI, hsvROI, sl, sh, hl, hh, vl, vh)
```

### #mencari piksel paling terang pada citra

```
minVal, maxVal, minLoc, maxLoc = most_bright(outputROI)
xv = maxLoc[0]
yv = maxLoc[1]
aaa = xv * factor_x_2
bbb = yv * factor_y_2
```



```

y_pos = [aaa,bbb]

if aaa != 0 or bbb != 0:
    y_pos = [aaa,bbb]
    pyautogui.moveTo(y_pos[0],y_pos[1])

    #jika push button 2 move dilepas
    if (last_received != "2"):
        flag = 0;
        pyautogui.mouseUp()
        break

#menampilkan frame, frameROI, dan maskingImage
cv2.imshow('frame',frame)
cv2.imshow('frameROI',frameROI)
cv2.imshow('frameMasking', outputROI)
##          print flag

cv2.circle(black,(int(width_video/2),int(height_video/2)), 1, (0,0,0), -1)

#menampilkan frame, frameROI, dan maskingImage
##          cv2.imshow('draw', black)
cv2.imshow('frame', frame)
cv2.imshow('frameROI', frameROI)
cv2.imshow('frameMasking', outputROI)

#jika q ditekan, program berhenti
if cv2.waitKey(1) & 0xFF == ord('q'):
    break

#jika q ditekan, program berhenti
if cv2.waitKey(1) & 0xFF == ord('p'):
    pyautogui.click()

#seluruh windows ditutup
cv2.destroyAllWindows()

```

## Lampiran 1.2 Program *input move* dan *drag* pada Arduino

```

const int buttonPinMove = 7;
const int buttonPinClick = 6;
const int ledPin = 13;
int buttonStateMove = 0;
int buttonStateClick = 0;
void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPinMove, INPUT);
  pinMode(buttonPinClick, INPUT);
  digitalWrite(buttonPinMove, HIGH);
  digitalWrite(buttonPinClick, HIGH);
}

void loop(){
  buttonStateMove = digitalRead(buttonPinMove);
  buttonStateClick = digitalRead(buttonPinClick);
  if (buttonStateMove == LOW) {
    digitalWrite(ledPin, HIGH);
    Serial.println("1");
  }
  else if (buttonStateClick == LOW) {
    digitalWrite(ledPin, HIGH);
    Serial.println("2");
  }
  else {
    Serial.println("0");
    digitalWrite(ledPin, LOW);
  }
}

```

## DAFTAR RIWAYAT HIDUP



**Muhamad Aldiansyah** merupakan putera sulung dari Ibu Rodiah dan Bapak Mansyur yang lahir di Tangerang, 24 Mei 1995.

Pendidikan formal dimulai dari SDN Kapuk 15 Pagi Jakarta (2001-2005), SDN Periuk 3 Tangerang (2005-2007), SMPN 12 Tangerang (2007-2010), SMAN 1 Kabupaten Tangerang (2010-2013). Semasa kuliah pernah menjadi asisten laboratorium Elektronika dan Fisika Komputasi, peserta olimpiade OSN-Pertamina, dan menjadi pemakalah Seminar Nasional Fisika 2017 dalam bidang instrumentasi dan komputasi di Universitas Negeri Jakarta. Pernah magang di Lembaga Perlindungan Anak Pandawa Care. Pernah bekerjasama dengan Badan Meteorologi Klimatologi dan Geofisika (BMKG) untuk melaksanakan Praktik Kerja Lapangan pada tahun 2016 dalam bidang pengolahan citra satelit.

Di luar bidang akademik, penulis pernah aktif dalam organisasi Art of Physics Jurusan Fisika UNJ sebagai Ketua pada tahun 2016 – 2017 serta menjadi Koordinator Tim Seni Fisika UNJ pada tahun 2017.