

# Sistem Pengecekan Lembar Jawaban Komputer Dengan *Optical Mark Recognition (OMR)* Berbasis *Open Computer Vision Python*

## *Computer Answer Sheet Checking System With Optical Mark Recognition (OMR) Based-On Open Computer Vision Python*

Iqlima Zahari<sup>1\*</sup>, Zudha Pratama<sup>2</sup>, Wildan Mahmud<sup>3</sup>, Dibyo Adi Wibowo<sup>4</sup>

<sup>1,3</sup>Sistem Informasi, Universitas Dian Nuswantoro

<sup>2,4</sup>Teknik Informatika, Universitas Dian Nuswantoro

Email: <sup>1</sup>iqlima.zahari@dsn.dinus.ac.id, <sup>2</sup>zudhapratama@dsn.dinus.ac.id, <sup>3</sup>wildan.mahmud@dsn.dinus.ac.id,

<sup>4</sup>dibyoadiwibowo@dsn.dinus.ac.id

\*Penulis korespondensi: [iqlima.zahari@dsn.dinus.ac.id](mailto:iqlima.zahari@dsn.dinus.ac.id)

### ABSTRAK

Sistem pengecekan lembar jawaban komputer sudah tidak asing lagi digunakan pada saat ujian pada lembaga pendidikan maupun pada sebuah lembaga non pendidikan. Umumnya metode yang digunakan pada saat mengecek lembar jawab computer adalah dengan menggunakan metode OMR (Optical Mark Recognition). Umumnya teknologi OMR ini mahal dan menjadikan OMR penggunaannya terbatas bagi instansi tertentu. Alternatif yang dapat kita gunakan adalah teknologi computer vision yang dapat kita manfaatkan dari beberapa pustaka opensource dan OpenCV salah satunya. Penulis mencoba membangun sistem pengecekan LJK alternatif OMR menggunakan OpenCV dalam bahasa Python. Selain membangun sistem pengecekan LJK, penulis juga melakukan pengujian untuk mengetahui konsistensi akurasi nilai yang dihasilkan sistem dengan beberapa skenario uji. Pengujian menggunakan cara hasilnya dibandingkan dengan pengecekan manual oleh manusia. Hasilnya sistem pengecekan LJK tersebut mampu konsisten menghasilkan nilai yang akurat selama kondisi LJK tidak terlalu miring, tidak terlalu terang atau terlalu gelap untuk pencahayaannya.

**Kata kunci:** optical mark recognition, OpenCV, python

### ABSTRACT

The computer answer sheet checking system is no stranger to being used during exams at educational institutions or at a non-educational institution. Generally, the method used when checking computer answer sheets is to use the OMR (Optical Mark Recognition) method. Generally, OMR technology is expensive and makes OMR use limited for certain institution. An alternative that we can use is computer vision technology which we can use from several open source libraries and OpenCV is one of them. The author tries to build an alternative OMR LJK checking system using OpenCV in Python. In addition to building the LJK checking system, the author also conducted tests to determine the consistency of the accuracy of the values generated by the system with several test scenarios. The test uses the way the results are compared with manual checking by humans. As a result, the LJK checking system is able to consistently produce accurate values as long as the LJK conditions are not too tilted, not too bright or too dark for the lighting.

**Keywords:** optical mark recognition, OpenCV, python.

## 1. PENDAHULUAN

Pengecekan lembar jawaban komputer menggunakan LJK (lembar jawab komputer) merupakan metode memasukkan data ke dalam komputer tanpa menggunakan keyboard. Cara ini telah banyak digunakan sejak tahun 1970-an oleh sekolah, pemerintah, industri, penyedia layanan kesehatan dan dalam pemasaran untuk

berbagai alasan seperti tes, survei pelanggan, riset pasar, pemilihan umum, pengujian laboratorium dan banyak lagi. Ini menunjukkan jika penggunaan LJK sudah sejak lama digunakan sebagai alat untuk meringankan pekerjaan manusia dalam memasukkan data ke komputer. Namun ada satu permasalahan sistem pengecekan dengan LJK yakni membutuhkan biaya tinggi dalam pembuatannya.

Permasalahan yang utama pada penggunaan LJK adalah biaya yang tinggi untuk membuat sistem tersebut dan perlu membuat sistem pengecekan LJK dengan harga yang lebih rendah. Harga yang relatif tinggi ini karena komponen penyusun sistem tersebut lumayan rumit dan mahal. alternatif nya adalah menggantikan komponen yang rumit tersebut dengan metode lain yang mampu menghasilkan keluaran yang sama. Misalnya kita bisa melibatkan kecerdasan buatan dari penyedia bersifat terbuka atau *open source*.

Kecerdasan buatan merupakan kemampuan sistem untuk menafsirkan data eksternal dengan benar, untuk belajar dari data tersebut, dan menggunakan pembelajaran tersebut untuk mencapai tujuan dan tugas tertentu melalui adaptasi yang fleksibel [1]. Teknologi informasi yang melibatkan kecerdasan buatan, big data, perangkat Internet of Things, dan blockchain telah dikembangkan dan diimplementasikan di banyak bidang teknik di seluruh dunia [2]. Salah satu cabang kecerdasan buatan yang dapat membantu kita dalam menyelesaikan permasalahan pengecekan LJK adalah *Computer Vision*

*Computer Vision* adalah metode untuk memperoleh, memproses, menganalisis dan memahami gambar digital [3], dan mengekstraksi data berdimensi tinggi dari dunia nyata untuk menghasilkan informasi numerik atau simbolis [4]. Secara singkatnya dapat kita artikan pemrosesan sedemikian rupa sebuah gambar untuk memperoleh nilai yang dapat digunakan untuk membuat keputusan. Hal tersebut sejalan dengan cara kerja sistem pengecekan LJK yang mengambil data atau informasi dari visual dokumen (data digital dokumen). Salah satu pustaka yang menyediakan pemrosesan *computer vision* yang *open source* adalah OpenCV.

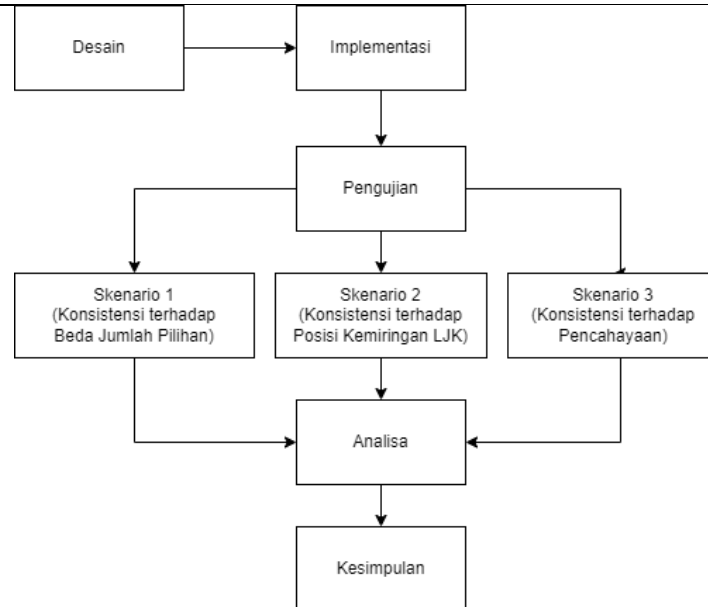
Penelitian terkait penggunaan computer vision pertama adalah OpenCV untuk penginderaan terhadap meteran listrik, digunakan untuk membaca digit angka meteran [5]. OpenCV akan melakukan pemrosesan gambar untuk mendapatkan nilai tersebut. Sistem akan mengolah pixel pada gambar yang didapatkan dan melakukan proses learning untuk setiap karakter/digit yang ada dalam gambar tersebut. Perbedaannya dengan proses pengecekan LJK adalah yang di baca pada LJK hanya area arsir, tanda silang atau coretan apapun pada kolom dalam LJK yang disediakan. Berikutnya beberapa artikel penelitian dalam negeri yang menggunakan OpenCV untuk pengecekan LJK. Pertama terdapat artikel publikasi yang menggunakan OpenCV untuk pengecekan LJK pada aplikasi berbasis android dengan bahasa java [6]. Berikutnya pengecekan LJK menggunakan image recognition menggunakan pustaka EmguCV [7]. Prinsip kerjanya hampir sama namun memiliki perbedaan pada bahasa pemrograman yang digunakan. Pada beberapa jurnal internasional juga dibahas pengecekan LJK menggunakan OpenCV yang menggunakan pemrosesan gambar dengan proses canny dan deteksi *contour* (pola) [8], kurang spesifik pola yang seperti apa yg dideteksi. Berikutnya pengecekan LJK menggunakan OpenCV, dengan pemrosesan gambar menggunakan proses grayscale kemudian dilakukan deteksi terhadap *contour* lingkaran [9].

Berdasarkan penelitian-penelitian diatas terlihat jika pengecekan LJK dapat dilakukan dengan beberapa pemrosesan gambar seperti *grayscale*, *canny*, *image recognition*, pencarian *contour*, bahkan spesifik ke *contour* lingkaran. Disana juga tertulis penggunaan beberapa pustaka yang tersedia bisa menggunakan OpenCV (java dan python) serta dapat juga menggunakan pustaka EmguCV.

Penulis memanfaatkan OpenCV dalam sistem pengecekan LJK dan menggunakan kamera webcam. Pemrosesan foto LJK menggunakan beberapa tahap pemrosesan gambar antara lain *grayscale*, *blur*, *canny*, *biggest contour*, *perspective transform*, *color threshold* dan *segmentation grading*. Selain membuat sistem tersebut penulis juga melakukan pengujian untuk memastikan konsistensi nilai yang dihasilkan, Hasil pengecekan sistem akan diuji dengan beberapa skenario untuk melihat keakuratan dan kemampuan sistem ketika posisi LJK sedikit miring dan dijalankan pada pencahayaan yang berbeda.

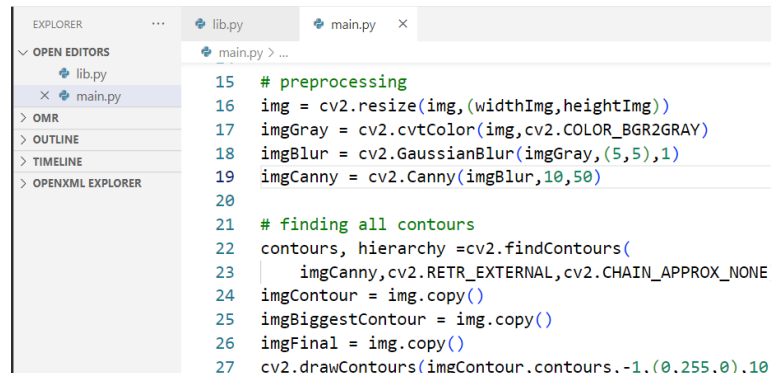
## 2. METODE PENELITIAN

Metode dalam penelitian ini menggunakan metode eksperimen. Metode eksperimen merupakan suatu cara untuk melakukan pengujian tentang sesuatu hal, mengamati prosesnya serta mencatat hasil pengujiannya, kemudian hasil tersebut di evaluasi, agar diperoleh hasil yang optimal. Berikut tahapan atau prosedur penelitian yang dilakukan.



**Gambar-1.** Diagram metode penelitian dengan 3 skenario pengujian

- Desain: tahap awal dimana desain *flowchart* yang dapat digunakan untuk panduan dalam penyusunan baris kode program dapat dengan cepat dipahami, pustaka eksternal apa saja yang dibutuhkan dan fungsi-fungsi apa saja yang dipersiapkan pada saat menyusun program untuk membuat sistem pengecekan LJK.
- Implementasi: Penulisan baris kode python sesuai desain yang dibuat. Setiap bagian didefinisikan dengan menuliskan baris kode sesuai tahapan pemrosesan gambar, mulai dari *preprocessing*, mencari *contour*, menentukan *contour* yang dipilih, *perspective transform*, *threshold*, *segmentation* dan *grading*.



```

EXPLORER
...
lib.py
main.py x
OPEN EDITORS
lib.py
main.py
OMR
OUTLINE
TIMELINE
OPENXML EXPLORER
main.py > ...
15 # preprocessing
16 img = cv2.resize(img, (widthImg, heightImg))
17 imgGray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
18 imgBlur = cv2.GaussianBlur(imgGray, (5, 5), 1)
19 imgCanny = cv2.Canny(imgBlur, 10, 50)
20
21 # finding all contours
22 contours, hierarchy = cv2.findContours(
23     imgCanny, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)
24 imgContour = img.copy()
25 imgBiggestContour = img.copy()
26 imgFinal = img.copy()
27 cv2.drawContours(imgContour, contours, -1, (0, 255, 0), 10)
    
```

**Gambar-2.** Contoh implementasi baris kode untuk tahap preprocessing dan pencarian *contour*

- Pengujian: merupakan suatu proses uji dengan membuat beberapa skenario pengujian. Pengujian dilakukan dengan mengganti variabel bebas penelitian tergantung pada objek yang di uji. Pada penelitian kali ini kita akan menguji akurasi dan konsistensinya jika dibanding pengecekan manual. Dengan ketika jumlah baris pilihan lebih banyak, sudut kemiringan LJK terhadap kamera di ubah posisinya dan dengan pencahayaan yang berbeda-beda.
- Pengujian skenario 1: jumlah pilihan jawaban pada LJK yang berbeda. Pengujian tersebut dilakukan dengan beberapa macam LJK. Peneliti membuat LJK dengan baris berjumlah 5, 10, 15, 20 dan kelipatan berikutnya. Alasan pengujian adalah apakah jumlah baris akan berpengaruh pada hasil pengecekan LJK. Karena mengingat keterbatasan resolusi kamera yang berpengaruh pada kepadatan pixel dari gambar yang dihasilkan. Ada kemungkinan pengaruhnya karena ukuran bulatan pilihan jawaban makin kecil ketika dibuat makin banyak barisnya.
- Pengujian skenario 2: sudut kemiringan LJK terhadap kamera webcam dibuat berbeda. Pengujian tersebut dilakukan beberapa kali pencatatan, dan hanya menggunakan salah satu LJK. Variabel bebasnya adalah posisi LJK terhadap kamera yang dibuat berbeda-beda. Setiap pencatatan posisi atau jarak kamera tetap,

yang di berputar adalah LJK. Dimana LJK diputar searah jarum jam dengan beberapa sudut kemiringan. Sudut yang digunakan adalah 15, 30, 45 dan 60 derajat. Pengujian dilakukan karena ada dugaan jika kemampuan transformasi perspektif dari sistem bisa saja memiliki batasan pada sudut tertentu.

- d) Pengujian skenario 3: tingkat pencahayaan saat pengecekan LJK yang dibuat berbeda-beda. Pengujian tersebut sama seperti scenario dua yang hanya menggunakan satu LJK saja. Variabel bebas yang digunakan adalah kondisi cahaya saat pengecekan LJK dilakukan. Yakni kondisi cahaya sangat terang dengan menambahkan lampu tambahan didekat LJK, kondisi terang dengan hanya menggunakan cahaya lampu ruangan, kondisi redup ketika dilakukan di sisi sudut ruangan jauh dari lampu, dan kondisi sangat redup ketika di letakkan benda untuk menghalangi cahaya ke arah LJK secara langsung.
- e) Analisa: tahap membandingkan data pencatatan dari setiap skenario pengujian. Data hasil pencatatan dibandingkan apakah setiap pengujian, sistem mampu menghasilkan nilai yang konsisten dan masih tetap akurat. Apakah sudah dilakukan sesuai dengan harapan sebelum pengujian, sehingga sistem yang dibangun dapat dinyatakan konsisten akurat. Hasil analisa akan digunakan untuk menarik kesimpulan dan juga dijadikan dasar mencari kekurangan sistem sehingga dapat dijadikan dasar untuk penelitian selanjutnya.
- f) Kesimpulan: tahap menghubungkan hasil analisa dan menarik kesimpulan terkait konsistensi dan akurasi hasil pengecekan LJK dari sistem yang telah dibangun. Kesimpulan juga menjelaskan bagaimana kondisi sistem dapat menghasilkan nilai yang akurat dan kapan sistem tidak mampu menghasilkan nilai yang sesuai.

### 3. HASIL DAN PEMBAHASAN

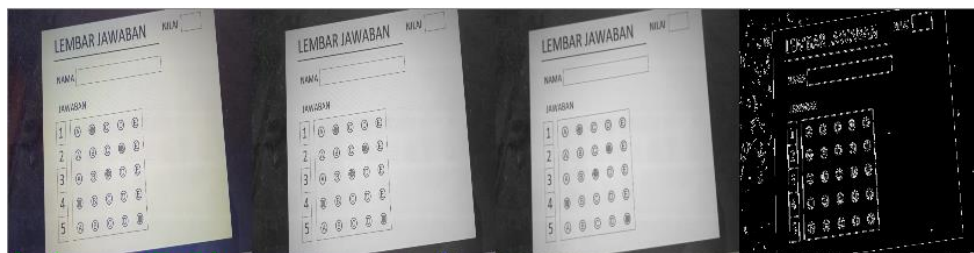
Pada bagian ini akan dipaparkan tentang pustaka penunjang serta tahapan-tahapan yang dilalui dalam penelitian sampai dihasilkan sebuah sistem pengecekan lembar jawab komputer menggunakan OpenCV python.

#### 3.1. Computer Vision

Kenneth Dawson & Howe menyebutkan jika *Computer Vision* merupakan sebuah analisis otomatis pada gambar dan video oleh komputer untuk mendapatkan suatu pemahaman. *Computer Vision* terinspirasi dari kemampuan sistem penglihatan manusia. Dengan kata lain, *Computer Vision* merupakan kemampuan penglihatan manusia yang diterapkan pada sebuah komputer [10]–[12]. Komputer memperoleh penglihatan melalui perangkat optic (misal: kamera webcam) yang menghasilkan video. Dari video tersebut diambil salah satu frame untuk dilakukan ekstraksi fitur pada untuk diambil nilai. Pemrosesan tersebut secara cepat dan iterative pada periode tertentu.

#### 3.2. Ekstraksi Fitur

Dalam *machine learning* dan *image processing*, ekstraksi fitur dimulai dari pemilihan dari kumpulan data awal terukur dan membangun nilai turunan (fitur) yang dimaksudkan agar informatif dan tidak berlebihan, memfasilitasi langkah pembelajaran dan generalisasi selanjutnya, dan dalam beberapa kasus dijadikan panduan untuk interpretasi manusia yang lebih baik. Ekstraksi fitur ini terkait dengan pengurangan dimensi agar data yang dihasilkan lebih spesifik [13]. Dalam pengecekan LJK ekstraksi fitur dimulai dari mengecilkan *range* pixel yang diproses dengan pemrosesan grayscale pada gambar, kemudian diproses Blur dan Canny untuk mendapat area tepi yang spesifik untuk mengenali beberapa area. Setelah mendapat beberapa area yang sesuai maka baru dilakukan pemrosesan lebih lanjut pengenalan pola pilihan jawaban terpilih.



Gambar-2. Preprocessing untuk ekstraksi fitur dengan metode grayscale, blur dan canny

### 3.3. Pencarian Biggest Contour

Merupakan tahap lanjutan setelah ekstraksi fitur berhasil menemukan beberapa pola (contour) namun masih perlu di pilah lagi pola mana yang akan diproses. Pada tahap ini kita perlu memberikan penanda urutan pola mana yang dipilih untuk diproses.



**Gambar-3.** Pencarian Biggest Contour

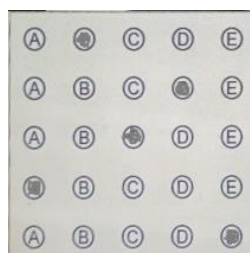
Jadi setiap pola akan diurutkan mulai dari terbesar ke terkecil. Kita perlu menentukan contour paling besar untuk digunakan misal untuk kolom identitas, kemudian contour terbesar kedua misal untuk kolom jawaban, dan seterusnya. Jadi untuk memberikan penanda area mana yang dipilih menggunakan perbandingan ukuran contour. Untuk membedakan kolom jawaban, nanti dibagi lagi dengan cara pembagian matrik kolom setelah antara area identitas dan jawaban dibagi.



**Gambar-4.** Contour yang sudah terpilih / tertandai

### 3.4. Perspective transform

Transformasi perspektif digunakan untuk mengubah konten tampilan layar sesuai ke posisi yang diinginkan pengamat, dan dapat membalik gambar dari sudut pandang yang berbeda[14]. Pada *contour area* yang memuat informasi jawaban sesuai gambar asli tangkapan kamera maka akan susah dibaca oleh sistem karena posisinya sulit untuk di pecah menjadi matrik. Solusi nya adalah dengan menerapkan transformasi perspektif yang seakan-akan gambar yang sebelumnya ditangkap kamera tampak miring seperti jajar genjang dapat menjadi lurus kotak persegi. Bentuk persegi hasil transformasi perspektif akan memudahkan program untuk mendapatkan matrik pilihan jawaban terpilih dari LJK.

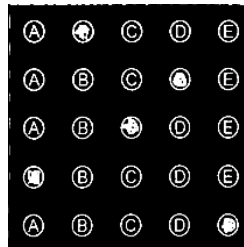


**Gambar-5.** Transformasi perspektif pada contour jawaban.

### 3.5. Threshold (ambang batas)

Tahap *threshold* akan mengubah perbedaan nilai pixel pada gambar menjadi lebih sederhana dengan menerapkan ambang batas nilai dari pixel gambar. Jika nilai piksel lebih besar dari nilai threshold, maka akan diberi suatu nilai (misal putih), jika tidak, diberi nilai lain (misal hitam) [15]. Sehingga perbedaan warna antara hitam dan putih akan semakin terlihat. Perbedaan yang semakin kontras ini akan mempermudah proses

pengambilan nilai perbedaan intensitas warna. Persentase kandungan warna putih antara kolom terpilih (yang berisi coretan) dan yang hanya menampilkan pilihan jawaban saja. Proses segmentasi matrik kolom jawaban menjadi cell kecil tiap jawaban akan dibahas pada tahap berikutnya.

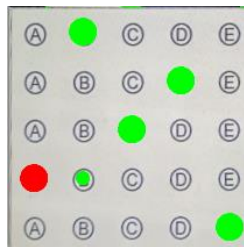


Gambar-6. Penerapan Threshold pada matrix jawaban

### 3.6. Segmentation and Grading

Proses Segmentasi adalah proses untuk memecah matrik jawaban menjadi kotak-kotak kecil seukuran jawaban. Setiap kotak jawaban hasil segmentasi akan dilihat kandungan pixel yang berwarna putih. Dalam satu baris akan terdapat beberapa kotak kecil jawaban, kotak-kotak tersebut akan dibandingkan kadar pixel putih tertinggi, kadar tertinggi tersebut maka akan menjadi jawaban terpilih dan ditampung dalam array.

Proses Grading adalah proses membandingkan array hasil perhitungan hasil segmentasi dengan array kunci jawaban. Jika nilai dalam array jawaban sama dengan nilai array kunci jawaban maka jawaban dianggap benar, jika tidak maka jawaban dianggap salah. Hasil perbandingan ini juga disimpan lagi ke dalam array hasil. Hasil grading akan di representasikan dalam sebuah titik berwarna tergantung benar dan salah nya. Jika benar maka titik diberi warna hijau, jika salah maka akan diberi warna merah seperti gambar 7.



Gambar-7. Hasil Segmentasi dan Grading

### 3.7. Pengujian Sistem

Pengujian sistem pengecekan LJK terbagi dalam 3 skenario penguji dengan satu tujuan, yakni untuk memastikan konsistensi akurasi hasil pengecekan. Kondisi seperti apakah yang sistem terima, agar sistem mampu memberikan hasil yang sesuai. Hasil sesuai yang dimaksud adalah hasil yang sama, jika dibandingkan pada pengecekan LJK secara manual.

a) Skenario 1 - uji dengan jumlah baris pilihan yang berbeda

Pada skenario uji pertama, hasil pengecekan menggunakan sistem yang sudah dibuat dengan pengecekan manual dilihat akurasi nya dan apakah hasilnya konsisten menghasilkan nilai yang sesuai atau tidak.

**Tabel -1.** Pengujian konsistensi hasil terhadap jumlah pilihan yang berbeda

Nomor uji	Jumlah baris pilihan	Hasil pengecekan (jumlah benar)	
		Manual	Sistem
1.1	5	4	4
1.2	10	9	9
1.3	15	13	13
1.4	20	18	18

Hasil pengujian diatas menunjukkan bahwa ketika jumlah baris dibuat lebih banyak, maka sistem akan konsiten memberikan nilai hasil pengecekan LJK yang konsisten. Hasil nya sama dengan pengecekan manual. Ketika jumlah semakin banyak maka pixel hasil segmentasi akan semakin kecil dan sistem berhasil mengecek

pilihan jawaban terpilih dan hasil sesuai.

b) Skenario 2 - uji terhadap posisi kemiringan yang berbeda

Pada skenario uji kedua, pengecekan menggunakan sistem yang dibuat dengan posisi kemiringan dokumen terhadap kamera di ubah pada beberapa derajat kemiringan yang berbeda dan dengan jumlah baris yang sama dengan sengaja salah satu jawaban dibuat salah. Apakah nilai yang dihasilkan sistem dengan pengecekan manual menunjukkan nilai yang konsisten.

**Tabel -2.** Pengujian konsistensi hasil terhadap posisi kemiringan yang berbeda

Nomor uji	Kemiringan LJK terhadap kamera	Hasil pengecekan (jumlah benar)	
		Manual	Sistem
2.1	15 derajat	9	9
2.2	30 derajat	9	9
2.3	45 derajat	9	9
2.4	60 derajat	9	-

Hasil pengujian diatas menunjukkan bahwa ketika posisi LJK terhadap kamera dibuat semakin miring maka masih tetap menghasilkan hasil yang sesuai hingga sudut 45 derajat. Namun ketika sudut dibuat lebih miring maka sistem gagal memberikan nilai. Kemungkinan proses perspective transform mengalami kegagalan sehingga proses pembuatan matriks jawaban pada segmentasi dan hal tersebut berdampak pada proses grading yang akhirnya juga mengalami kegagalan.

c) Skenario 3 - uji terhadap pencahayaan yang berbeda

Pada skenario uji ketiga, pengecekan menggunakan sistem dengan kondisi cahaya yang berbeda. LJK yang digunakan masih sama dengan skenario kedua dan dengan sengaja salah satu jawaban dibuat salah. Saat pengecekan LJK dihadapkan ke kamera akan diberikan pencahayaan yang sangat terang, terang, redup, dan sangat redup. Apakah nilai yang dihasilkan sistem dengan pengecekan manual menunjukkan nilai yang konsisten atau tidak.

**Tabel -3.** Pengujian konsistensi hasil terhadap pencahayaan yang berbeda

Nomor uji	Pencahayaan	Hasil pengecekan (jumlah benar)	
		Manual	Sistem
3.1	Sangat Terang	9	-
3.2	Terang	9	9
3.3	Redup	9	9
3.4	Sangat Redup	9	8

Hasil pengujian diatas menunjukkan bahwa ketika pencahayaan pada LJK dibuat sangat terang sistem gagal memberikan nilai. Ketika cahaya terang dan redup masih memberikan hasil sesuai. Dalam pengujian ini kondisi redup adalah hanya menggunakan lampu ruangan, tanpa lampu webcam. Dan ketika dibuat semakin gelap atau sangat redup maka sistem menghasilkan nilai yang kurang sesuai. Sistem tidak mampu mengidentifikasi pilihan jawaban ketika pencahayaan kurang.

#### 4. KESIMPULAN

Berdasarkan hasil pengujian yang dilakukan maka dapat ditarik kesimpulan bahwa sistem pengecekan LJK yang dibangun dengan OpenCV Python mampu memberikan hasil yang diharapkan. Hasil pengecekan tetap akurat walau dengan jumlah baris pilihan LJK yang berbeda-beda namun dengan beberapa kondisi terpenuhi. Sistem akan tetap menghasilkan nilai yang sesuai jika sudut nya tidak terlalu miring, tidak terlalu terang dan tidak terlalu gelap. Penelitian ini masih menggunakan satu kamera untuk pengujian nya dan mungkin beberapa kekurangan yang ditemukan di atas dapat diselesaikan jika menggunakan kamera yang lebih bagus resolusinya. Atau bisa saja kamera dengan kemampuan *night view* yang lebih baik. Sehingga perlu di uji lagi dengan beberapa kamera yang memiliki kemampuan yang baik.

**UCAPAN TERIMA KASIH**

Penulis menyampaikan terima kasih kepada Lembaga Penelitian dan Pengabdian Kepada Masyarakat (LPPM) Universitas Dian Nuswantoro yang telah mendukung penuh penulis dalam mengikuti agenda seminar nasional ini.

**DAFTAR PUSTAKA**

- [1] A. M. Kaplan and M. Haenlein, "Siri, Siri, in my hand: Who's the fairest in the land? On the interpretations, illustrations, and implications of artificial intelligence," *Bus. Horiz.*, 2019.
- [2] J. Li, M. S. Herdem, J. Nathwani, and J. Z. Wen, "Methods and applications for Artificial Intelligence, Big Data, Internet of Things, and Blockchain in smart energy management," *Energy AI*, vol. 11, p. 100208, 2023, doi: <https://doi.org/10.1016/j.egyai.2022.100208>.
- [3] R. Klette, *Concise Computer Vision: an introduction into theory and algorithms*. Springer London, 2014.
- [4] F. Hu, H. Tang, A. Tsema, and Z. Zhu, "Chapter 1 - Computer Vision for Sight: Computer Vision Techniques to Assist Visually Impaired People to Navigate in an Indoor Environment," in *Computer Vision for Assistive Healthcare*, M. Leo and G. M. Farinella, Eds. Academic Press, 2018, pp. 1–49.
- [5] S. Sivkov *et al.*, "The algorithm development for operation of a computer vision system via the OpenCV library," *Procedia Comput. Sci.*, vol. 169, pp. 662–667, 2020, doi: <https://doi.org/10.1016/j.procs.2020.02.193>.
- [6] Y. I. Hernafi, T. A. Riza, and H. Hafidudin, "Aplikasi Android Koreksi Lembar Jawaban Komputer Menggunakan Opencv," *eProceedings Appl. Sci.*, vol. 6, no. 1, pp. 447–489, 2020, [Online]. Available: <https://openlibrarypublications.telkomuniversity.ac.id/index.php/appliedscience/article/view/11793>.
- [7] D. A. Sinuraya and M. S. Hasibuan, "Pemeriksa Lembar Jawaban dengan Image Recognition," *J. Informatics Telecommun. Eng.*, vol. 2, no. 1, pp. 1–8, Jul. 2018, doi: 10.31289/jite.v2i1.1496.
- [8] A. E. ZekiKüçükkara, "An Image Processing Oriented Optical Mark Recognition and Evaluation System," *Int. J. Appl. Math. Electron. Comput.*, vol. 6, no. 4, pp. 59–64, 2018.
- [9] P. Raundale, T. Sharma, S. Jadhav, and R. Margaye, "Optical Mark Recognition using Open {CV}," *Int. J. Comput. Appl.*, vol. 178, no. 37, pp. 9–12, Aug. 2019, doi: 10.5120/ijca2019919093.
- [10] K. Dawson-Howe, *A Practical Introduction to Computer Vision with OpenCV*. John Wiley & Sons, 2014.
- [11] V. G. Dhanya *et al.*, "Deep learning based computer vision approaches for smart agricultural applications," *Artif. Intell. Agric.*, vol. 6, pp. 211–229, 2022, doi: <https://doi.org/10.1016/j.aiia.2022.09.007>.
- [12] M. Holba, P. Bilik, and M. Kelnar, "Image Processing in Defectoscopy," *IFAC-PapersOnLine*, vol. 48, no. 4, pp. 65–70, 2015, doi: <https://doi.org/10.1016/j.ifacol.2015.07.009>.
- [13] S. Sarangi, M. Sahidullah, and G. Saha, "Optimization of data-driven filterbank for automatic speaker verification," 2020, doi: 10.48550/ARXIV.2007.10729.
- [14] J. Zhang *et al.*, "A perspective transformation method based on computer vision," in *2020 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, 2020, pp. 765–768, doi: 10.1109/ICAICA50127.2020.9182641.
- [15] J. Munoz-Minjares, O. Vite-Chavez, J. Flores-Troncoso, and J. M. Cruz-Duarte, "Alternative Thresholding Technique for Image Segmentation Based on Cuckoo Search and Generalized Gaussians," *Mathematics*, vol. 9, no. 18, 2021, doi: 10.3390/math9182287.