Journal of System and Computer Engineering (JSCE) ISSN (online): 2723-1240
Vol. 4 No. 1 Januari 2023

Aplikasi Penghitung Kapasitas Ruang Parkir pada Lahan Parkir Kosong Menggunakan Library OpenCV pada Bahasa Pemrograman Python

Devi Ayu Lestari [1], Mochamad Wisuda Sardjono [2], Maulana Mujahidin[3]
Email : [1]deviayulestari240@gmail.com, [2]moch_wisuda@staff.gunadarma.ac.id,
[3]mujahidin@staff.gunadarma.ac.id

Universitas Gunadarma

Abstrak

Dalam penelitian ini, penulis membahas tentang pembuatan aplikasi penghitung ketersediaan lahan parkir kosong menggunakan metode *Image Processing.* Lahan parkir area terbuka dijadikan objek penelitian. OpenCV merupakan library pada bahasa pemrograman Python yang dibutuhkan selama proses *Image Processing* berlangsung. OpenCV membantu untuk membuat aplikasi yang berkaitan dengan citra seperti gambar ataupun foto dan video. Pada aplikasi ini, teknologi yang digunakan adalah video *image processing* melalui perekaman video yang diambil dari sudut atas menggunakan kamera drone. Rekaman video pergerakan mobil dalam area parkir dibutuhkan pada aplikasi ini. Kemudian data tersebut diolah melalui sebuah *engine*, lalu *output* yang dihasilkan dari aplikasi berupa file gambar dari sebuah lahan parkir yang sedang memasuki ataupun meninggalkan lahan parkir dalam bentuk format file.png. Program aplikasi memberi nilai dari setiap *object* yang berada pada lahan parkir, jika nilai *object* < 900 maka tidak ada *object* kendaraan pada lahan parkir tersebut dan jika nilai *object* > 900 maka terdapat *object* kendaraan pada lahan parkir. Berdasarkan nilai *object* tersebut maka dilakukan perhitungan lahan parkir kosong dan lahan parkir yang terisi. Dalam tahap uji coba dengan menggunakan sampel berupa video dengan 10 responden menyatakan 92.6% aplikasi ini dapat berfungsi sesuai tujuannya dan ditemukan tidak ada masalah dalam perhitungan pada aplikasi serta aplikasi dapat berjalan dengan baik dan seperti tujuan dalam pembuatannya yaitu sebagai penghitung ketersediaan area parkir pada suatu lahan parkir terbuka.

1. Pendahuluan

Lahan parkir merupakan kondisi ruang untuk menempatkan kendaraan yang tidak bergerak dan memiliki sifat penempatannya sementara. Tempat parkir merupakan prasarana transportasi yang tidak dapat dipisahkan dari sistem jaringan transportasi, sehingga dengan pengaturan parkir mempengaruhi kinerja suatu jaringan pada lahan parkir (Kurniawan & Sriharyani, 2018). Dengan sifat penempatan yang sementara maka diperlukan pengaturan penempatan yang maksimal untuk memanfaatkan lahan parkir. Tantangan yang terjadi pada saat mengelola lahan parkir area terbuka adalah jumlah kapasitas kendaraan yang terparkir dan parkir kosong membutuhkan waktu untuk menghitungnya.

Di era modern saat ini telah menunjukkan adanya perkembangan teknologi informasi yang terjadi disetiap waktu dengan harga terjangkau. Kini telah banyak hadir teknologi-teknologi canggih sehingga banyak membantu manusia dalam kegiatan sehari-hari (Ashqer & Bikdash, 2019). Salah satu pengembangan teknologi yaitu dalam bidang transportasi yang banyak ditemui seperti fitur yang digunakan untuk

menganalisa jumlah kendaraan yang keluar masuk pada area parkir terbuka. Seringnya terjadi kendala dalam mencari area parkir yang kosong pada suatu area parkir yang luas dengan mengelilingi area parkir minimal satu kali sehingga kurang efisien dan membutuhkan waktu yang lama. Penggunaan sistem parkir berbasis komputer dapat memudahkan dalam pencatatan terhadap data area parkir yang kosong. Selain itu dengan adanya sistem parkir berbasis komputer, keamanan di area parkir akan lebih terjaga.

Visi komputer *(computer vision)* merupakan bidang yang menjadi tantangan menarik untuk membuat komputer dapat menangkap informasi yang berada dalam suatu gambar atau video. Beberapa penelitian mengenai *Image processing* sebagai salah satu bagian dalam *computer vision* saat ini sangat berkembang dengan pesat (Listartha, Indrawan, & Pramartha, 2020) (Lopez, Griffin, Ellis, Enem, & Duhan, 2019). Pada aplikasi ini *Image processing* sangat membantu dalam pengolahan citra gambar atau video dalam menghitung object kendaraan dan parkir kosong. Bahasa pemrograman Python dengan *library* OpenCV dilibatkan karena menyediakan modul-modul yang membuat *computer vision* menjadi mudah untuk digunakan (Kadir, 2019) (Rizkatama, Nugroho, & Suni, 2021) (Rosebrock, 2016). Pada pemrograman python terdapat beberapa *develepment tools* yang dapat digunakan salah satunya, PyCharm yang *userfriendly* dalam menggunakannya. (Nguyen, 2019)

Pada penelitian ini dibahas mengenai perancangan pembuatan aplikasi yang memberikan informasi jumlah ketersediaan lahan parkir kosong pada suatu area parkir dilapangan terbuka. Ukuran *object* kendaraan, lahan parkir, pengendara, pejalan kaki dan keranjang belanja menjadi acuan dalam menentukan lahan parkir tersebut kosong atau tidak. Format file gambar dan video lahan parkir dalam bentuk file **png** dan file **mp4** yang diambil menggunakan perangkat *drone*, sehingga area parkir dapat terlihat dari tampak atas. Salah satu dari 3 area parkir terdapat kondisi yang berbeda yaitu memiliki trotoar ditengah area parkiran kendaraan.

Langkah-langkah pembuatan aplikasi ini diawali dengan studi pustaka mengenai cara kerja library OpenCV dan *Image processing* dalam mendeteksi foto dan video sehingga hasil yang didapat sesuai dengan yang diinginkan. Penelitian mengenai *image processing* dengan *object* lahan parkir telah dilakukan oleh beberapa peneliti diantaranya dengan menggunakan CUDA (GPU) dan algoritma YOLO (Jupiyandi, Saniputra, Pratama, & Dharmawan, 2019), OpenCV dan Algortima YOLO v4 (Rizkatama, Nugroho, & Suni, 2021). Studi lapangan dilakukan dengan melihat secara langsung lahan parkir di setiap tempat yang berbeda-beda pada setiap pusat perbelanjaan.

Penelitian ini bertujuan untuk membangun sebuah aplikasi menggunakan Library OpenCV dalam menghitung ketersediaan lahan parkir kosong pada suatu area parkir terbuka dengan bahasa pemrograman Python. Lahan parkir dinyatakan penuh apabila jumlah *object* kendaraan sama dengan jumlah lahan parkir.

Dengan aplikasi ini dapat membantu pengendara yang ingin memarkirkan kendaraannya di suatu area parkir terbuka karena pengelola area parkir tersebut dapat menginformasikan jumlah area parkir sesuai dengan kondisi dilapangan. Membantu pengendara untuk mencari alternatif lahan parkir lain, jika lahan parkir dinyatakan penuh

2. Tinjauan Literatur

Penelitian yang berhubungan dengan deteksi ruang parkir dengan memanfaatkan perangkat Graphical Processing Unit (GPU) dan algoritma YOLO telah dilakukan dengan hasil waktu komputasi meningkat sebesar 0,179 detik dibandingkan dengan penggunaan CPU. Hal ini dipengaruhi oleh pemanfaatan GPU dalam mengolah citra pada *object* kendaraan dan lahan parkir. (Jupiyandi, Saniputra, Pratama, & Dharmawan, 2019)

Penggunaan dataset COCO dilakukan pada penelitian mengenai penghitungan jumlah mobil untuk mengetahui ketersediaan lahan parkir menggunakan Python dan YOLO v4. Pada penelitian ini dihasikan tingkat akurasi deteksi mobil sebagai penghitung ketersedaian lahan parkir sebesar 72,8%. (Rizkatama, Nugroho, & Suni, 2021)

Teknik *Gaussian Blur* digunakan pada penelitian *Internet of Thing*, pendeteksian lahan parkir berdasarkan pengolahan citra gambar. Pada penelitian ini digunakan algoritma *canny edge detection* dan bahasa pemrograman *Raspberry Pi*. (Listartha, Indrawan, & Pramartha, 2020)

Pendeteksian lahan parkir di area kampus menggunakan *live webcam* dengan variasi kondisi cuaca, pencahayaan yang berbeda dengan interval waktu setiap 10 menit. Metode *edge detection* digunakan untuk pendeteksian kendaraan. Terdapat perbedaan akurasi berdasarkan variasi kondisi cuaca dan pencahayaan. (Lopez, Griffin, Ellis, Enem, & Duhan, 2019)

Pendeteksian tempat parkir menggunakan image processing, sistem akan mendapatkan video streaming langsung melalui smartphone (browser) dari tempat parkir dari kamera, algoritma *Deep Neural Network* (DNN) digunakan untuk proses citra gambar. (Hattale, Hangam, Khilare, Ratnaparkhi, & Kasture, 2021)

Perpaduan *library OpenCV* dan metode deteksi objek statis, *Haar-like Cascade Classifier* dikombinasikan dengan *Hough Line Detection* digunakan untuk mengidentifikasi area parkir kosong pada citra gambar yang diambil secara *realtime* melalui kamera IP atau kamera USB. (Afriliana, Rosalina, & Valeria, 2018)

Pada penelitian ini digunakan metode *Gaussian Blur*, *Grayscale* dan *Adaptive Thresholding*, karena sebagian besar *image processing* menggunakannya dengan tingkat akurasi yang baik.

3. Metode Penelitian

Dalam bagian ini dijelaskan tentang perancangan dan implementasi dari Aplikasi penghitung kapasitas lahan parakir kosong menggunakan metode *Image Processing* dan library OpenCv pada Bahasa pemrograman Python berdasarkan metodologi SDLC yang sudah dijelaskan pada bab sebelumnya. Dimana aplikasi tersebut dijalankan dengan menggunakan laptop dengan mengolah data dari video.MP4 dan file.PNG yang sudah disediakan sebelumnya. Aplikasi akan menganalisis gambar dan video terkait ketersediaan parkir yang kosong. Hasil akhir pengolahan data dan analisis aplikasi tersebut yaitu berupa jumlah ketersediaan parkir yang kosong pada sebuah lahan parkir. Metode yang digunakan pada penelitian ini diantaranya *Gaussian blur, Grayscale* dan *Adaptive Threshold*.

Gaussian Blur merupakan konvolusi citra dengan fungsi *GaussianBlur()*. Istilah ini dibuat karena *Gaussian* memiliki beberapa sifat yang cukup unik. Pertama karena hasil transformasi *fourier* fungsi *Gaussian* ternyata menghasilkan *Gaussian* juga.

Kedua yang menarik dari fungsi *Gaussian* adalah untuk implementasi fungsi *Gaussian* 2D. Fungsi *Gaussian* 2D secara matematis memiliki sifat dapat dipisahkan (*separable*). *Gaussian blur* mirip dengan *Average blur* tetapi pada *Gaussian blur* menggunakan *weiggted mean*, yang akan membuat pixel disebelahnya lebih dekat ke pusat dan rata-rata pixel akan berkontribusi lebih banyak "*weight*". Hasilnya, *image* yang dihasilkan tidak terlalu kabur, tetapi memiliki tingkat blur secara alami daripada menggunakan metode *Average blur*. *Gaussian blur* didapat dari operasi konvolusi, dimana dilakukan perkalian antara matriks kernel dengan matriks gambar asli. Pada pelaksanaan konvolusi, kernel digeser sepanjang baris dan kolom dalam citra sehingga diperoleh nilai citra yang baru. (Ashqer & Bikdash, 2019) (Firdaus & Imelda, 2018) (Sonka, Hlavac, & Boyle, 2015)

Citra *grayscale* merupakan sebuah proses yang mengolah citra dengan mengubah nilai piksel pada cittra menjadi warna keabuan. Transformasi ini bertujuan untuk meningkatkan kontras pada citra, sehingga informasi-informasi pada citra bisa lebih terlihat. Warna yang merupakan tingkatan dari warna abu-abu hanya digunakan oleh citra *grayscale.* Tingkat keabuan disini yaitu warna abu-abu dari berbagai tingkat seperti dari hitam hingga yang mendekati putih. Warna abu-abu pada citra *grayscale* adalah warna R (*red*), G (*green*), B (*blue*) yang memiliki intensitas yang sama. Dalam *grayscale image* hanya membutuhkan nilai intensitas tunggal dimana intensitas dari citra *grayscale* disimpan dalam 8 *bit integer* yang memberikan 256 kemungkinan yang mana dimulai dari level 0 sampai dengan 255 (0 untuk hitam dan 255 untuk putih dan nilai diantaranya adalah derajat keabuan). (Pratt, 2001)

Thresholding adalah binarisasi dari sebuah *image* dan merupakan salah satu teknik segmentasi yang baik digunakan untuk citra dengan perbedaan nilai intensitas yang signifikan antara latar belakang dari objek utama. Biasanya *thresholding* digunakan untuk fokus pada suatu objek atau sare yang menarik dalam sebuah *image*. Dalam penerapannya, *thresholding* membutuhkan suatu nilai yang digunakan sebagai nilai pembatas antara objek utama dengan latar belakang dan nilai tersebut dinamakan *threshold.* (Habibah & Kurniawan, 2021) (Solomon & Breckon, 2011)

Berdasarkan teknik yang ada saat ini, *thresholding* dapat diklasifikasikan menjadi dua, yaitu *thresholding global* dan *thresholding* lokal (adaptif). *Thresholding* lokal bertujuan untuk menangani kesulitan yang disebabkan oleh intensitas variasi, dimana nilai *threshold* ditentukan dari setiap piksel berdasarkan nilai *grayscale* sendiri dan nilai *grayscale* tetangga. Oleh karena itu, pendekatan ini disebut *thresholding* adaptif. Nilai *threshold* T dihitung menggunakan rata-rata dan nilai intensitas citra. Proses adaptif *threshold* ini dilakukan untuk mentransformasi citra *grayscale* menjadi citra deteksi tepi (*edge detection*). (Habibah & Kurniawan, 2021) (Solomon & Breckon, 2011)

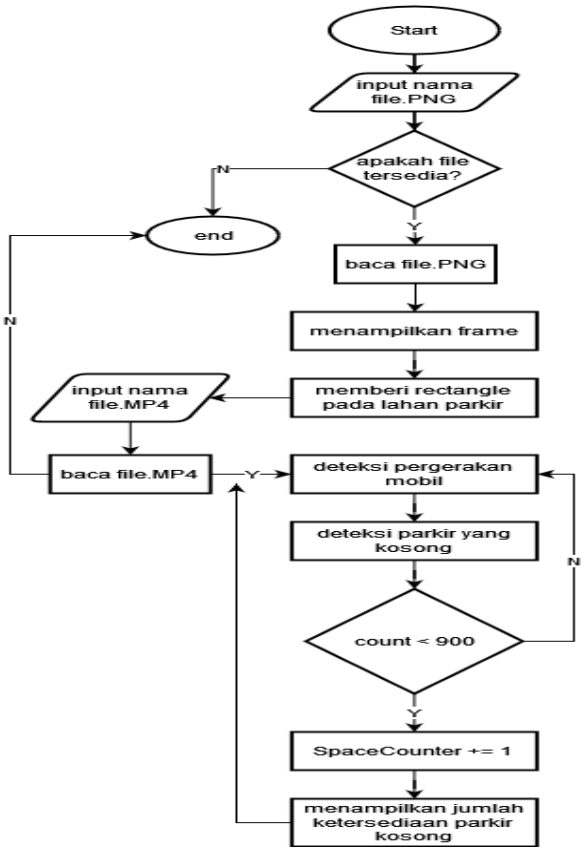
3.1. Diagram Alur

Diagram Alur merupakan salah satu bentuk algoritma yang dapat menjelaskan tahapan-tahapan cara kerja suatu sistem untuk memudahkan dalam memahami cara kerja sistem dapat dilihat pada Gambar 1.

Adapun langkah-langkah yang dijalankan berdasarkan diagram alur sebagai berikut:

1. Menginputkan nama file.PNG terlebih dahulu untuk memberi *rectangle* pada lahan parkir yang tersedia pada gambar.

2. Program akan mengecek apakah file.PNG tersebut tersedia atau tidak. Jika tersedia, maka program akan menampilkan *frame* dari gambar dan berlanjut untuk pemberian *rectangle* pada setiap lahan parkir. Dan jika tidak tersedia, maka program akan berhenti.
3. Jika sudah mendapatkan data lahan parkir, langkah selanjutnya pada file program yang kedua akan dilakukan penginputan nama file.MP4 lalu program akan membaca file yang sudah diinputkan.
4. Jika file.MP4 tidak terbaca, maka program akan berhenti. Jika file tersebut terbaca maka program akan mendeteksi pergerakan mobil pada setiap lahan parkir yang sudah diberi *rectangle*.
5. Program juga akan mendeteksi lahan parkir yang kosong ataupun yang sudah terisi oleh mobil.
6. Program akan memberi nilai dari setiap object yang berada pada lahan parkir yang tersedia. Jika Count < 900, itu berarti tidak ada object besar yaitu mobil pada lahan parkir tersebut. Jika count > 900, artinya terdapat object besar yaitu mobil pada lahan parkir.
7. Jika count < 900, maka program akan mendeteksi kembali pergerakan mobil dan melakukan perhitungan terhadap mobil yang masuk ataupun meninggalkan lahan parkir. Jika count > 900, maka program akan menambahkan 1 dari setiap lahan parkir yang memiliki nilai count > 900.
8. Program akan menampilkan jumlah ketersediaan lahan parkir kosong per jumlah keseluruhan dari lahan parkir yang tersedia.



Gambar 1. Diagram Alur

3.2. Perancangan Sistem

Tahap ini merupakan tahap dalam mengetahui apa yang menjadi kebutuhan pengguna aplikasi dalam menggunakannya dirumuskannya. Tahapan ini berperan penting dalam proses pendeteksian lahan parkir yang kosong. Pada sistem ini, pengguna adalah pengamat monitor yang memonitoring layar untuk mengetahui ketersediaan lahan parkir yang kosong.

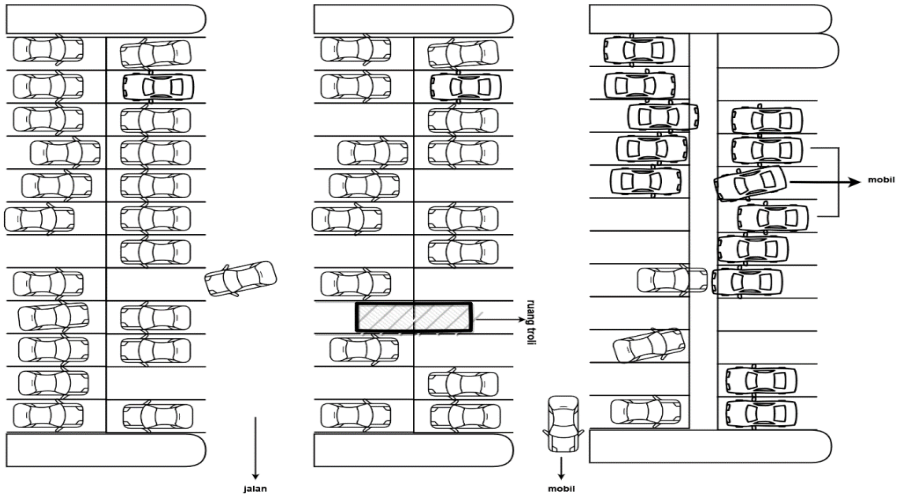
Berdasarkan data yang tersedia, didapatkan perencanaan untuk aplikasi pendeteksi lahan parkir kosong ini, yaitu:

1. Aplikasi dapat mendeteksi pergerakan mobil
   Aplikasi ini dapat melakukan serangkaian proses untuk mendeteksi dan menghitung pergerakan mobil saat hendak meninggalkan lahan parkir.
2. Aplikasi bersifat *input-based*
   Aplikasi ini melakukan pemrosesan dengan cara memasukkan video dan gambar yang telah direkam untuk mendapatkan data dan melakukan analisa.

Pada penelitian ini, data diambil dari 1 gambar dan 1 video. Data gambar digunakan untuk memberi *rectangle* pada setiap lahan parkir baik yang berisi mobil ataupun yang tidak terdapat mobil pada suatu lahan parkir tersebut. Data video digunakan untuk melakukan perhitungan dan pengecekan ketersediaan parkir dari pergerakan mobil pada saat memasuki ataupun meninggalkan lahan parkir. Pada data video diambil dari tampak atas sehingga akan terlihat jelas pergerakan mobil pada lahan parkir tersebut.

3.3. Perancangan Aplikasi

Pada aplikasi ini, teknologi yang digunakan adalah video *image processing* melalui perekaman video yang diambil dari sudut atas menggunakan kamera drone. Data yang dibutuhkan oleh aplikasi ini adalah rekaman video dari mobil saat memasuki ataupun meninggalkan lahan parkir. Kemudian data tersebut akan diolah melalui sebuah engine, lalu output yang dihasilkan dari aplikasi berupa file gambar dari sebuah lahan parkir yang sedang memasuki ataupun meninggalkan lahan parkir dalam bentuk format file.png. Alur proses yang terjadi secara umum dapat dilihat pada Gambar 2.



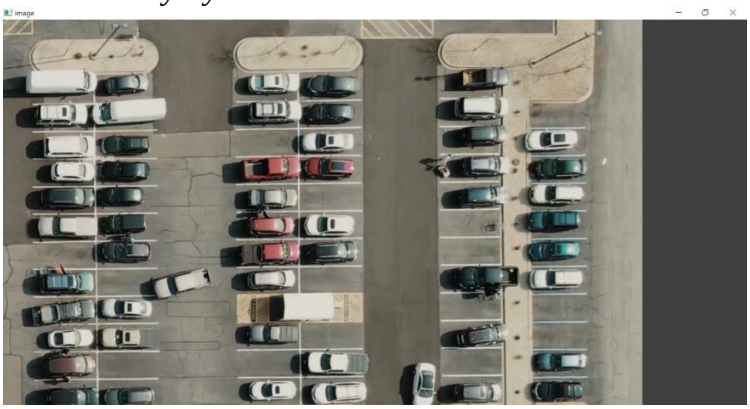
Gambar 2. posisi sudut (Angle) Kamera Saat Perekaman Video

Pada tahapan perancangan aplikasi ini dimulai denga merancang tampilan halaman aplikasi yang merupakan penghubung antara pengguna dan aplikasi.

4. Hasil dan Pembahasan

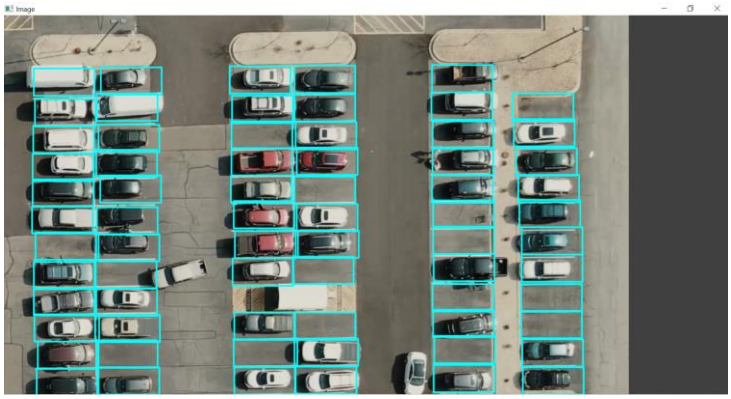
Setelah melakukan tahapan perancangan aplikasi, tahapan berikutnya adalah membuat pengkodean program menggunakan bahasa pemrograman Python dengan *development tools* PyCharm. OpenCV digunakan sebagai salah satu *library tools*.

Untuk tahapan pertama pada pembuatan kode program yaitu dengan membaca atau mengimport citra gambar lahan parkir dengan format file **PNG**. Pickel digunakan untuk menyimpan data ke dalam sebuah file, yang merupakan salah satu modul pada *standard library Python*.



Gambar 3. Citra Lahan Parkir tampak atas (drone)

Pada Gambar 3. ditunjukan citra gambar lahan parkir tampak dari atas menggunakan perangkat drone. Setelah itu citra lahan parkir tersebut ditandai posisi parkir terebut dengan membuat ukuran *rectangle* yaitu menentukan ukuran *width* dan *height*.



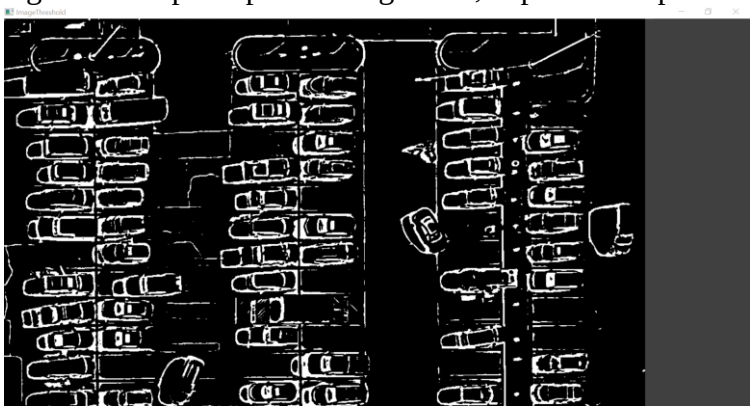
Gambar 4. *Rectangle* lahan parkir

Citra gambar lahan parkir tampak atas yang didapatkan dari *drone*, awalnya berupa citra video yang sebelumnya telah dinyatakan pada kode program pada saat me*load* citra video. Sehingga tidak diperlukan menggambar ulang *rectangle* lahan parkir.

Setelah terbentuk *rectangle* lahan parkir, dilakukan pemrosesan gambar dengan membuat citra gambar lahan parkir tersebut menjadi *blur* dengan kernel kecil 3x3 dan sigma x=1, menghasilkan binerisasi citra gambar dan pengaburan *Gaussian* menggunakan *grayscale* dengan spesifikasi COLOR_BGR2GRAY, ditampilkan pada Gambar 4.

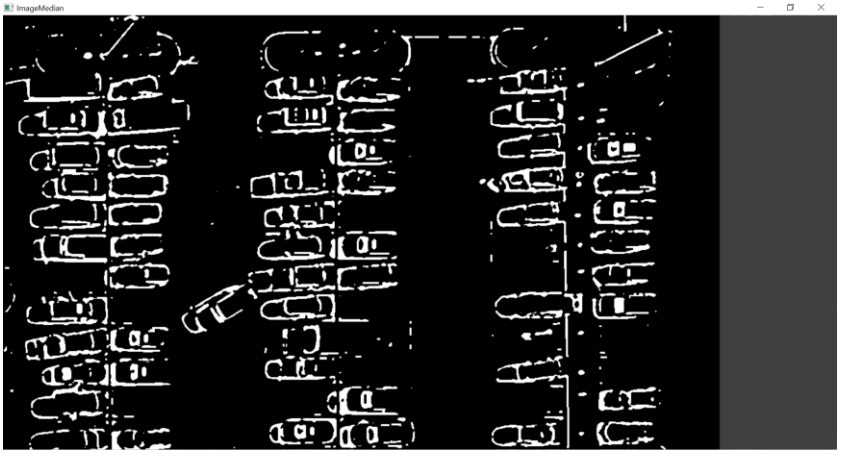
Gambar 4. *Grayscale* dan *Bluring*

Thresholding digunakan untuk memfokuskan objek atau area pada citra gambar. Penggunaan *Gaussian blur* pada *adaptive Thresholding* sangat membantu untuk menghilangkan *noise* putih pada citra gambar, dapat dilihat pada Gambar 5.



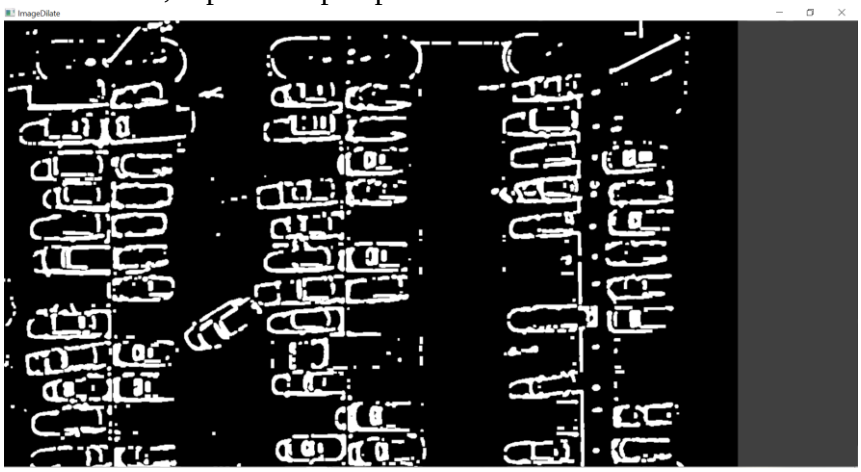
Gambar 5. *Adaptive Thresholding*

Selanjutnya membuat pengkodean program *median blur* untuk meningkatkan nilai pixel pada garis putih agar lebih mudah membedakan lahan parkir yang sudah terisi ataupun yang masih kosong. Ketika tidak ada kendaraan di tempat parkir ditunjukan dengan tidak adanya atau sedikitnya pixel putih pada citra gambar yang sudah dilakukan proses *median blur*, tampak pada Gambar 6.



Gambar 6. Proses *Median Blur*

Image Dilate digunakan untuk membuat garis pada pixel putih menjadi lebih tebal sehingga lebih mudah untuk membedakan lahan parkir yang kosong ataupun yang sudah terisi mobil, seperti tampak pada Gambar 7.



Gambar 7. Proses *Image Dilate*

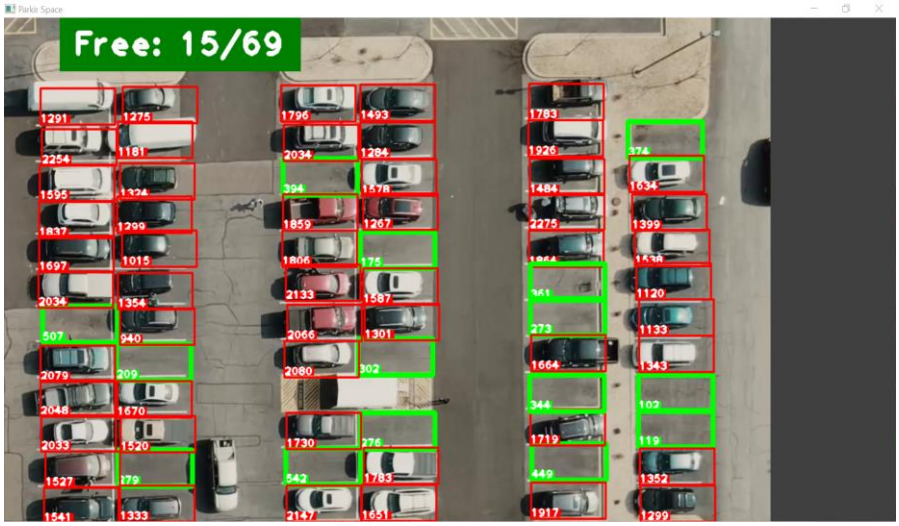
Langkah berikutnya membuat kode program untuk menentukan apakah tempat parkir sudah terisi atau belum terisi oleh mobil. Program akan mendeteksi kendaraan yang masuk ataupun keluar dengan perbandingan nilai pada *rectangle.* Apabila nilai < 900, maka *rectangle* akan berwarna hijau yang berarti lahan tersebut kosong. Program akan melakukan penjumlahan +1 untuk ketersediaan parkir kosong. Sebaliknya, apabila nilai > 900, maka *rectangle* akan berwarna merah yang berarti lahan parkir sudah terisi oleh kendaraan lain. Kode perintah dapat dilihat pada Gambar 8.

```
def checkParkingSpace(imgPro):
    spaceCounter = 0
    for pos in posList:
        x,y = pos
        imgCrop = imgPro[y:y+height,x:x+width]
        #cv2.imshow(str(x*y),imgCrop)
        count = cv2.countNonZero(imgCrop)

        if count < 900:
            color = (0,255,0)
            thickness = 5
            spaceCounter += 1
        else:
            color = (0,0,255)
            thickness = 2
        cv2.rectangle(img, pos, (pos[0] + width, pos[1] + height), color, thickness)
        cvzone.putTextRect(img,str(count),(x,y+height-3), scale=1,
                           thickness=2, offset=0, colorR=color)
```

Gambar 8. Kode perintah lahan parkir terisi atau tidak oleh kendaraan

Pada Gambar 9. dapat dilhat jumlah total dari keseluruhan lahan parkir yaitu 69 dan jumlah ketersediaan lahan parkir kosong adalah 15. *Rectangle* hijau menandakan bahwa lahan tersebut merupakan lahan parkir kosong dan *rectangle* merah menandakan bahwa lahan parkir tersebut sudah terisi oleh kendaraan. Perhitungan ketersediaan lahan parkir kosong akan terus berubah sesuai dengan kendaraan yang menempati area yang sudah dijadikan lahan parkir.



Gambar 9. Jumlah lahan parkir

Aplikasi ini telah dilakukan pengujian dan implementasi. Perangkat keras yang digunakan dalam membuat aplikasi yaitu menggunakan komputer atau laptop dengan spesifikasi:

a. *Processor* Intel Core i5-8265U
b. RAM 12 GB
c. *Solid State Drive* 512 GB
d. Sistem Operasi Windows 10 64-bit

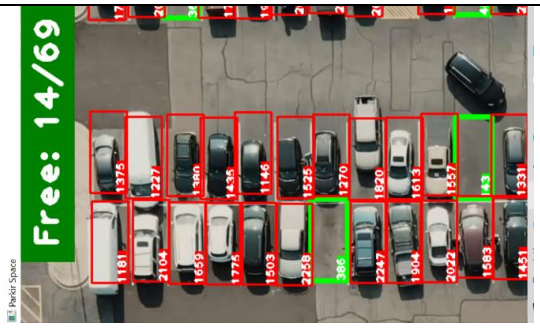
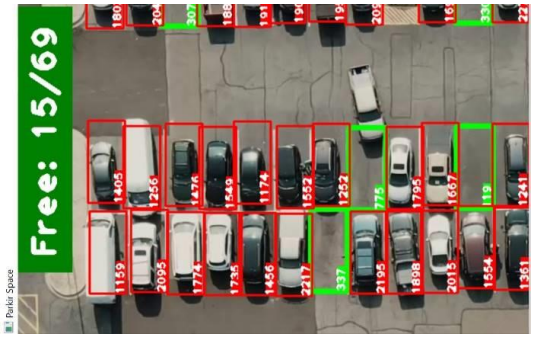
Sedangkan untuk perangkat lunak yang digunakan dalam membuat aplikasi ini diantaranya:

a. OpenCV versi 4.6.0.66
b. Bahasa Pemrograman Python versi 3.10
c. PyCharm community edition 2021.3.3

Pengujian merupakan tahap yang utama dalam pembuatan suatu aplikasi. Pengujian ini dilakukan untuk mengetahui hasil yang didapat dari aplikasi yang telah dibuat. Hasil dari pengujian, akan dijadikan sebagai tolak ukur dalam pengembangan selanjutnya.

Pada pengujian aplikasi penghitung lahan parkir yang kosong menggunakan OpenCV dan metode *image processing*, bahan uji coba yaitu video yang diambil dari tampak atas. Uji coba dilakukan untuk mengetahui apakah semua fungsi yang telah ditulis didalam program aplikasi berjalan sesuai dengan kebutuhan fungsional yang telah didefinisikan di dalam program tersebut dapat dilihat pada Tabel 1.

Tabel 1. Uji Coba Aplikasi

No.	Tampilan Percobaan	Keterangan
1		Aplikasi menampilkan bahwa hanya tersedia 2 parkir kosong pada lahan parkir bagian ujung kiri. Diketahui jumlah ketersediaan lahan parkir dari seluruh bagian yaitu 14/69.
2		Aplikasi menampilkan bahwa jumlah ketersediaan lahan parkir bertambah 1 menjadi 15/69. Hal ini terjadi karena terdapat 1 mobill yang hendak keluar meninggalkan lahan parkir sehingga program akan menghitung sebagai lahan parkir kosong.

3		Aplikasi berjalan dengan baik karena mampu mendeteksi objek kecil (manusia) dan tidak menghitungnya sebagai kendaraan yang memasuki lahan parkir. Hal itu dapat dilihat pada parkir bagian kanan ujung, terlihat manusia yang sedang membawa *trolley* dan melewati lahan parkir yang kosong.

Kesimpulan hasil uji coba pada aplikasi menggunakan sampel video adalah aplikasi berjalan dengan baik sesuai dengan tujuan dari dibuatnya aplikasi tersebut yaitu menghitung jumlah ketersediaan lahan parkir yang kosong.

Pengujian aplikasi juga dilakukan dengan menggunakan metode *User Acceptance Testing* (UAT). *User Acceptance Testing* (UAT) merupakan pengujian yang ditunjukkan untuk menghasilkan bukti bahwa aplikasi yang telah dibuat dapat diterima oleh penggunanya. Proses uji coba dengan menggunakan metode *User Acceptance Testing* (UAT) ini dilakukan dengan memberikan 10 pertanyaan kepada 10 responden. Tabel 2. merupakan hasil dari uji coba aplikasi dengan metode *User Acceptance Testing* (UAT).

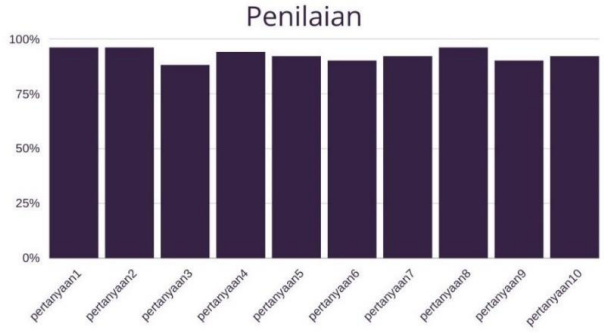
Tabel 1. Kuesioner UAT

NO	Pertanyaan	Jumlah						
		SS	S	R	TS	STS	Jml	%
1.	Apakah "Aplikasi Penghitung Ketersediaan Parkir Kosong" sudah berjalan dengan baik?	40	8	0	0	0	48	96
2.	Apakah aplikasi sudah memiliki perhitungan yang tepat untuk menghitung lahan parkir kosong?	40	8	0	0	0	48	96
3.	Apakah Aplikasi sudah dapat membedakan antara objek kecil (manusia) dan objek besar (mobil) dalam pendeteksiannya?	25	16	3	0	0	44	88
4.	Apakah aplikasi sudah dapat membedakan antara lahan parkir yang kosong dengan lahan parkir yang terisi oleh mobil?	35	12	0	0	0	47	94
5.	Apakah fitur-fitur yang ditampilkan pada aplikasi mudah untuk dimengerti?	35	8	3	0	0	46	92
6.	Apakah hasil dari aplikasi yang ditampilkan bisa untuk dimengerti?	25	20	0	0	0	45	90
7.	Apakah aplikasi sudah layak untuk digunakan secara publik?	35	8	3	0	0	46	92
8.	Apakah aplikasi akan mempermudah dalam penghitungan parkir kosong pada	40	8	0	0	0	48	96
9.	Apakah respon dari aplikasi saat mendeteksi lahan parkir yang kosong/terisi sudah cepat?	25	20	0	0	0	45	90

10.	Apakah aplikasi tergolong fleksibel untuk menentukan area mana yang layak dijadikan lahan parkir?	35	8	3	0	0	46	92

Keterangan :

- SS – Sangat Setuju (bobot: 5)
- S – Setuju (bobot: 4)
- R – Ragu Ragu (bobot: 3)
- TS – Tidak Setuju (bobot: 2)
- STS – Sangat Tidak Setuju (bobot: 1)



Gambar 10. Diagram Penilaian

Dari hasil analisa data yang dikumpulkkan dengan menggunakan metode UAT dari 10 responden, maka dapat disimpulkan dengan persentase 92.6% menyatakan aplikasi ini berjalan dengan baik, memiliki perhitungan yang tepat, dapat membedakan objek besar (mobil) dan objek kecil (manusia), fitur-fitur yang ditampilkan mudah dimengerti, sudah layak digunakan secara publik, mempermudah dalam perhitungan parkir kosong, sudah cepat dalam mendeteksi parkir kosong, dan tergolong fleksibel untuk menentukan area yang layak dijadikan lahan parkir.

5. Kesimpulan

Dari penilaian dan pembahasan “Aplikasi Penghitung Ketersediaan Lahan Parkir Kosong Menggunakan Library OpenCV Pada Bahasa Pemrograman Python“ dapat ditarik kesimpulan yaitu aplikasi yang dibangun dapat membantu pengguna yakni admin pada suatu area parkir untuk menghitung dan mengetahui ketersediaan parkir kosong pada suatu lahan parkir terbuka. Aplikasi juga sudah dapat membedakan antara objek kecil (manusia) dan objek besar (mobil) sehingga, objek kecil yang melintas pada lahan parkir kosong tidak akan terhitung sebagai mobil. Aplikasi yang dibuat juga sudah tergolong fleksibel karena pengguna (admin) bisa menghapus ataupun menambah area yang layak dijadikan lahan parkir sesuai dengan kebutuhan. Dalam pembuatan aplikasi ini digunakan bahasa pemrograman Python dan diimplementasikan menggunakkan library OpenCV. Dalam tahap uji coba dengan menggunakan sampel berupa video dengan 10 responden menyatakan 92.6% aplikasi ini dapat berfungsi sesuai tujuannya dan ditemukan tidak ada masalah dalam perhitungan pada aplikasi serta aplikasi dapat berjalan dengan baik dan seperti tujuan dalam pembuatnnya yaitu sebagai penghitung ketersediaan area parkir pada suatu lahan parkir terbuka.

Daftar Pustaka

Afriliana, N., Rosalina, & Valeria, R. (2018). PENDETEKSIAN RUANG KOSONG PARKIR DI DALAM RUANGAN. *Ultima Computing*, 34-40.

Ashqer, M. I., & Bikdash, M. (2019). PARKING LOT SPACE DETECTION BASED ON IMAGE PROCESSING. *Conference Proceeding IEEE SouthEastCon.*

Firdaus, A., & Imelda. (2018). PENERAPAN METODE GAUSSIAN BLUR DAN ABSOLUTE DIFFERENCE PADA JUMLAH DAN KECEPATAN KENDARAAN. *Skanika*, 1003-1011.

Habibah, M. U., & Kurniawan, M. (2021). SEGMENTASI CITRA WAJAH DENGAN IMPLEMENTASI ADAPTIVE THRESHOLD-INTEGRAL IMAGE. *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*, 919-928.

Hattale, P., Hangam, V., Khilare, S., Ratnaparkhi, a., & Kasture, P. (2021). PARKING SPACE DETECTION USING IMAGE PROCESSING. *International Journal of Science and Research (JSR)*, 1440-1442.

Jupiyandi, S., Saniputra, F. R., Pratama, Y., & Dharmawan, M. R. (2019). PENGEMBANGAN DETEKSI CITRA MOBIL UNTUK MENGETAHUI JUMLAH TEMPAT PARKIR MENGGUNAKAN CUDA DAN MODIFIED YOLO. *Jurnal Teknologi Informasi dan Ilmu Komputer (KTIIK)*, 413 - 419.

Kadir, A. (2019). *LANGKAH MUDAH PEMROGRAMAN OPENCV DAN PYTHON*. Jakarta: PT. ElexMedia Komputindo.

Kurniawan, S., & Sriharyani, L. (2018). *ANALISIS PENGARUH PARKIR DI BADAN JALAN TERHADAP KINERJA JALAN JENDERAL AHMAD YANI KOTA METRO*. Lampung: Tapak.

Listartha, I. M., Indrawan, G., & Pramartha, I. G. (2020). IOT - PARKING LOT DETECTION BASED ON IMAGE PROCESSING. *Jurnal Sistem dan Informatika (JSI)*, 168-176.

Lopez, M., Griffin, T., Ellis, K., Enem, A., & Duhan, C. (2019). PARKING LOT OCCUPANCY TRACKING THROUGH IMAGE PROCESSING. *Proceeding of 34th International Conference on Computers and Their Applications* (pp. 265-270). Hawaii, USA: Epic Computing.

Nguyen, Q. (2019). *HANDS-ON APPLICATION DEVELOPMENT WITH PYCHARM: ACCELERATE YOUR PYTHON APPLICATION USING PRACTICAL CODING TECHNIQUES IN PYCHARM*. Birmingham: Packt Publishing.

Pratt, W. K. (2001). *DIGITAL IMAGE PROCESSING: PIKS INSIDE, THIRD EDITION*. California: John Wiley and Son, Inc.

Rizkatama, G. N., Nugroho, A., & Suni, A. F. (2021). SISTEM CERDAS PENGHITUNG JUMLAH MOBIL UNTUK MENGETAHUI KETERSEDIAAN LAHAN PARKIR BERBASIS PYTHON DAN YOLO v4. *Edu Komputika Journal, Unnes*, 91-99.

Rosebrock, A. (2016). *PRACTICAL PYTHON AND OPENCV (3rd EDITION)*. PyImageSearch.

Solomon, C., & Breckon, T. (2011). *FUNDAMENTAL OF DIGITAL IMAGE PROCESSING: A PRACTICAL APPROACH WITH EXAMPLE IN MATLAB*. West Sussex: John Wily and Sons, Inc.

Sonka, M., Hlavac, V., & Boyle, R. (2015). *IMAGE PROCESSING, ANALYSIS, AND MACHINE VISION FOURTH EDITION*. USA: Cengage Learning.