# Sistem Pendeteksi Paket Pengiriman Rapuh dengan Segmentasi Citra

13517034 Muhammad Fariz Luthfan Wakan
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: luthfan.wakan@gmail.com

***Abstrak*— Aktivitas daring bukan merupakan hal yang asing lagi, terutama di era pandemi COVID-19. Peningkatan aktivitas daring selama transisi era pandemi COVID-19, memberikan kesegaran pada perusahaan ekspedisi (layanan pengiriman barang). Namun, *human error* dalam proses pengiriman barang semakin meningkat, salah satunya saat pensortiran barang yang perlu perhatian khusus (*fragile*). Tidak sedikit pelanggan yang melayangkan surat terbuka atas kekecewaannya di laman Google karena barangnya yang rusak. Oleh karena itu diperlukan otomasi dalam hal ini untuk mencegah kerugian yang dirasakan kedua belah pihak. Pada makalah ini akan dibahas mengenai pensoritan paket pengiriman rapuh otomasi dengan menggunakan segmentasi citra.**

**Kata kunci—COVID-19, fragile, segmentasi citra**

## I. PENDAHULUAN

Di zaman yang modern ini, aktivitas daring bukanlah merupakan hal yang asing, apalagi di era pandemi COVID-19, era New Normal, seperti kegiatan belajar mengajar yang dilaksanakan melalui *video conferencing*, *work from home* (WFH), dan lainnya. Hal ini disebababkan, di era ini, aktivitas daring dijadikan prioritas utama demi menjaga kesehatan dan mencegah terpaparnya virus yang tersebar di lingkungan luar, sehingga tidak keluar rumah selama sepekan sudah menjadi hal yang umum di kalangan masyarakat.

Dengan adanya peningkatan aktivitas daring, persentase belanja secara daring pun meningkat hingga 26% selama pandemi COVID-19[1]. Shopee, yang merupakan salah satu *online marketplace* di Indonesia, melaporkan terjadi peningkatan transaksi sebesar 130% selama Q2 2020[2].

Peningkatan persentase yang tinggi pada belaja daring menyebabkan bisnis ekspedisi atau jasa pengiriman barang menerima dampak yang sangat positif. JNE, yang merupakan salah satu perusahaan yang memberikan layanan pengiriman barang, mengalami pertumbuhan sebesar 30% di tengah pandemi[3]. Pertumbuhan sebesar itu dapat menjadi mimpi buruk bagi perusahaan ekspedisi, jika mereka tidak meningkatkan *customer experience* (CX) mereka. Jika dilihat dari laman Google, tidak sedikit pelanggan yang mengeluhkan masalahnya secara terbuka, salah satunya kerusakan barang[4].

Kerusakan barang saat pengiriman dapat disebabkan oleh beberapa hal, namun umumnya disebabkan oleh *human error*. *Human error* mungkin terjadi saat pensortiran barang yang perlu perhatian khusus (seperti barang *fragile*) dilakukan. Kesalahan ini dapat merugikan kedua belah pihak, baik perusahaan maupun pelanggan. Perusahaan perlu mengganti rugi barang pelanggan yang telah dirusaknya, sedangkan pengguna akan kehilangan waktu dan barangnya meskipun terdapat kompensasi.

Oleh karena itu, pensortiran barang yang memerlukan perhatian khusus perlu diotomasi untuk mencegah kerugian ini. Segmentasi citra dapat memiliki peranan yang penting dalam menyelesaikan permasalahan tersebut.

## II. DASAR TEORI

### A. Segmentasi Citra

Segmentasi citra adalah proses pembagian citra digital ke dalam beberapa bagian (objek). Segmentasi citra bertujuan untuk menemukan bagian citra yang koheren atau objek spesifik. Citra disegmentasi berdasarkan properti yang dipilih; seperti, warna, tekstur, dan lainnya. Berikut beberapa pendekatan dalam melakukan segmentasi citra.

*1)* Pendekatan *discontiuity*, pendekatan ini mempartisi citra berdasarkan perubahan nilai intensitas pixel yang cepat, seperti tepi (*edge detection*).

*2)* Pendekatan *similarity*, pendekatan ini mempartisi citra berdasarkan kemiripan area menurut properti yang ditentukan. Berikut metode segmentasi citra dengan pendekatan ini.

*a)* Pengambangan (*thresholding*), metode segmentasi citra yang didasarkan pada nilai intensitas pada pixel dan nilai ambang yang ditentukan (*threshold*). Metode ini menghasilkan citra biner dengan 2 (dua) daerah, yaitu daerah yang dianggap sebagai latar belakang dan bukan latar belakang. Pixel akan dianggap sebagai latar belakang jika nilai intensitas pada pixel tersebut lebih kecil dari nilai ambang, dan sebaliknya.

*b) Region growing*, metode segmentasi citra dengan memilih suatu kelompok pixel untuk dijadikan sebagai daerah awal yang nantinya akan tumbuh menjadi daerah yang lebih besar dengan melakukan pengecekan terhadap pixel–pixel di sekitarnya (pixel tetangga).

*c) Split and merge*, metode segmentasi citra yang menggunakan algoritma *divide and conquer*. Citra akan dibagi (*split*) menjadi sejumlah region yang tidak saling berhubungan

(*disjoint*), dan kemudian menggabungkan region-region bertetangga yang homogen.

*d) Clustering*, metode segmentasi citra dengan membentuk kelompok-kelompok berdasarkan homogenitas pixel-pixel tetangga. Segmentasi dengan *clustering* dapat dilakukan dengan mengubah setiap pixel di dalam citra menjadi vektor-vektor; intensitas, warna, dan warna-koordinat. Algoritma *clustering* sendiri memiliki variasi yang cukup beragam, seperti *k-means* dan *agglomerative*.

### B. *Edge Detection (pendeteksian tepi)*

Sebuah *edge* menandakan pinggiran dari sebuah objek. *Edge detection* bertujuan untuk meningkatkan penampakan garis batas atau objek di dlaam citra. Pendeteksian tepi mengekstraksi representasi gambar garis-gari di dalam citra. Oleh karena itu, pendeteksian tepi sanagt berguna dalam mengenali objek di dalam citra (image recognition). *Edge detection* dilakukan dengan mengidentifikasi pixel yang mengalami perubahan nilai intensitas yang cepat. *Edge detection* dapat dilakukan dengan menggunakan metode operator Canny, operator Prewitt, operator Sobel, LoG (Laplace of Gaussian), operator Robert, atau lainnya.

### C. *Operator Canny*

Operator Canny merupakan salah satu metode dalam melakukan pendeteksian tepi. Metode ini merupakan salah satu metode yang efisien dan sering digunakan dalam mendeteksi tepi. Salah satu karakteristik yang menjadi incaran banyak orang adalah bagaimana metode ini dapat menghasilkan tepi hanya dengan ketebalan 1 piksel saja. Pada prosesnya, operator Canny akan memilih suatu pixel sebagai tepi apabila besar gradien dari piksel tersebut lebih dari besar gradien dari piksel lain yang ada di sebelah piksel tersebut.

### D. *Model Warna (color space)*

Warna adalah spektrum tertentu yang terdapat di dalam suatu cahaya dengan identitasnya ditentukan oleh panjang gelombang cahaya tersebut. Segmentasi citra dapat dilakukan berdasarkan warna pada citra. Warna pada citra digital dapat direpresentasikan dalam bentuk model warna. Model warna yang paling umum dikenal adalah model warna RGB; yang terdiri dari *red* (merah), *green* (hijau), dan *blue* (biru). Model warna lainnya adalah CMYK, YCbCr, HIS, HSV, dan lainnya.

### E. *Hue Saturation Value (HSV)*

Selain model warna YCbCr, model warna merupakan model warna yang paling mendekati sistem visual manusia. Pada HSV, H (*hue*) merupakan jenis warna sebenarnya (seperti merah, ungu, dan lainnya) yang memiliki rentang nilai 0 s.d. $2\pi$, S (*saturation*) merupakan kemurnian warna yang memiliki rentang nilai [0, 1], dan V (*value*) merupakan kecerahan sebuah warna yang memiliki rentang nilai [0, 1].

dapat memiliki peranan yang penting dalam menyelesaikan permasalahan tersebut.

## III. Implementasi Segmentasi Citra pada Pendeteksian Paket Pengiriman Rapuh

OpenCV dan Python digunakan sebagai kakaks untuk mengimplementasikan segmentasi citra pada pendeteksian paket pengiriman rapuh, OpenCV dan Python digunakan sebagai sebagai kakas.

Sistem pendeteksi paket pengiriman rapuh dengan segmentasi citra terdiri dari 2 (dua) fase, sebagai berikut.

1. Pendeteksian paket, fase ini ditujukan unuk mendeteksi semua paket yang ada di dalam citra. Fase ini dilakukan dengan metode pendeteksian tepi.
2. Pendeteksian rapuh (*fragile*), fase ini ditujukan untuk mendeteksi apakah sebuah paket merupakan paket yang rapuh atau tidak rapuh. Fase ini dilakukan dengan memanfaatkan label *fragile* yang umumnya dipasangkan pada paket pengiriman rapuh, seperti pada Gambar III.1, dengan mendeteksi warna ciri khas pada label tersebut, yaitu merah.


**Gambar III.1** Label fragile pada paket pengiriman rapuh

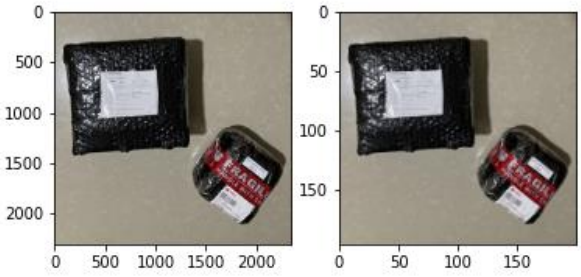
### A. *Mengubah Ukuran Citra Masukan*

Gambar yang telah diperoleh dari sumber akan dilakukan perubahan ukuran citra terlebih dahulu agar sesuai dengan harapan saat melakukan pemrosesan citra selanjutnya, terutama saat melakukan penyaringan kontur pada citra. Dengan dilakukannya tahap ini, ukuran citra tidak menjadi batasan pada sistem ini.

Pada OpenCV di Python, mengubah ukuran citra dilakukan dengan memanggil fungsi, sebagai berikut.

```
cv.resize(img, dim, intrpl)
```

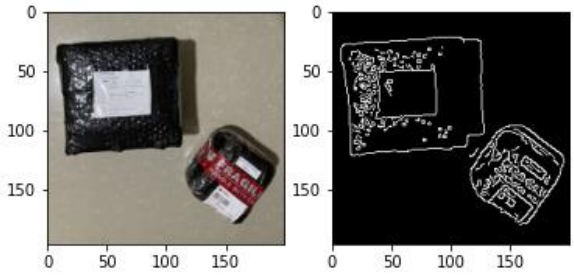
`img` merupakan citra masukan, `dim` merupakan ukuran yang diharapkan, dan `intrpl` merupakan metode interpolasi yang diinginkan dalam mengubah ukuran citra. Metode interpolasi `INTER_AREA` dipilih karena gambar masukan akan diperkecil.



**Gambar III.2** Perbandingan citra asli (kiri) dengan citra hasil *resize* (kanan)

*B. Mendeteksi Tepi dengan operator Canny*

Operasi ini merupakan operasi pertama pada fase pertama. Citra yang ukurannya telah diubah akan dilakukan pendeteksian tepi dengan operator Canny, sehingga dapat menghasilkan tepi objek yang memiliki ketebalan 1 (satu) pixel.



**Gambar III.3** Perbandingan citra masukan (kiri) dengan citra hasil operator Canny (kanan)

Pada OpenCV di Python, operator Canny diterapkan dengan memanggil fungsi, sebagai berikut.

**`cv.Canny(img, low_thld, high_thld)`**

`img` merupakan citra masukan, `low_thld` merupakan batas bawah deteksi tepi, dan `high_thld` merupakan batas atas deteksi tepi. Pada sistem ini, `low_thld` bernilai 140 dan `high_thld` bernilai 255.

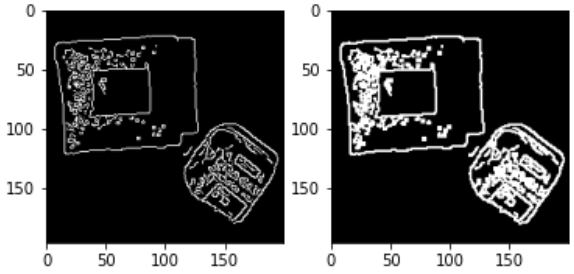
*C. Mendilatasi Citra Deteksi Tepi*

Citra yang telah diterapkan operator Canny akan dilakukan dilasi agar tepi citra menjadi lebih tebal. Ini dilakukan dalam upaya untuk menghubungkan kontur tepi yang kemungkinan putus pada satu objek.

Pada OpenCV di Python, dilatasi pada citra dilakukan dengan memanggil fungsi, sebagai berikut.

**`cv.dilate(img, kernel, iter)`**

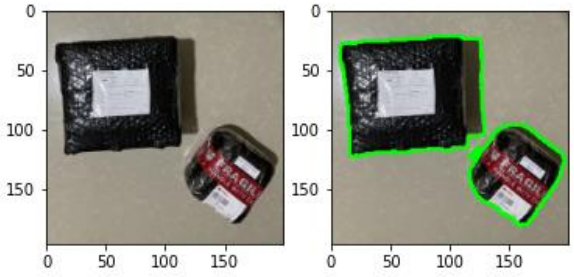
`img` merupakan citra masukan, `kernel` merupakan kernel konvolusi matriks, dan `iter` merupakan jumlah iterasi dilatasi yang dilakukan. Pada sistem ini, `kernel` berupa matriks 2 x 2 yang bernilai 1 dan iterasi dilakukan 1 kali (`iter` = 1) agar penebalan tidak terlalu intens.



**Gambar III.4** Perbandingan citra masukan (kiri) dengan citra hasil *dilate* (kanan)

*D. Mencari Kontur Citra*

Setelah dilasi dilakukan pada citra, operasi pencarian kontur dilakukan untuk menemukan tepi dari sebuah objek.



**Gambar III.5** Perbandingan citra asli (kiri) dengan citra hasil pencarian kontur (kanan)

Pada OpenCV di Python, pencarian kontur pada citra dilakukan dengan memanggil fungsi, sebagai berikut.

**`cv.findContours(img, mode, method)`**

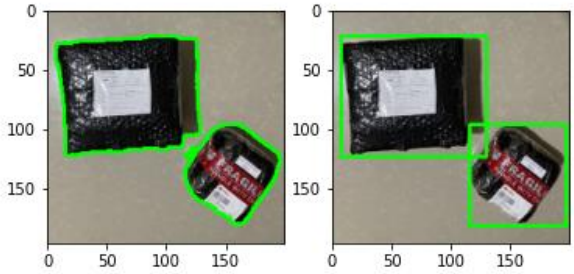
`img` merupakan citra masukan, `mode` merupakan mode dalam pencarian kontur, dan `method` merupakan metode yang digunakan dalam mencari kontur. Pada sistem ini, `mode` `RETR_EXTERNAL` dipilih agar menghasilkan kontur bagian luar saja dan `method` `CHAIN_APPROX_SIMPLE` dipilih untuk menyederhanakan hasil pencarian kontur.

*E. Menyaring Kontur Citra*

Jika dilihat pada gambar sebelumnya, Gambar III.5, citra hasil terapan kontur menghasilkan 3 (tiga) region, yang mana terdapat 1 (satu) region yang tidak diharapkan. Untuk menghilangkan region tersebut, dilakukan penyaringan kontur pada citra.

Pada sistem ini, penyaringan kontur dilakukan dengan melakukan pemanggilan fungsi **`cv.boundingRect`** terlebih dahulu untuk menghasilkan region dalam bentuk segi empat,

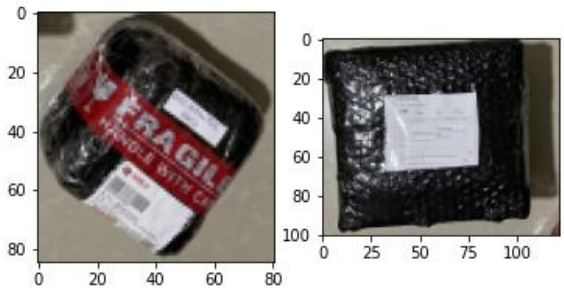
yang kemudian dilakukan penyaringan region berdasarkan luas region. Region yang akan tetap ada merupakan region yang memiliki luas 2000 pixel persegi.



**Gambar III.6** Perbandingan citra masukan (kiri) dengan citra hasil *boundingRect* dan penyaringan kontur (kanan)

*F. Mengekstraksi Objek*

Setelah region objek telah dihasilkan, operasi pertama pada fase kedua dilakukan, yaitu melakukan ekstraksi objek. Eksraksi dilakukan dengan memotong gambar menggunakan *x-axis* (`x`), *y-axis* (`y`), *width* (`w`), dan *height* (`h`) yang dihasilkan dari pemanggilan fungsi **`cv.boundingRect`**.



**Gambar III.7** Citra objek hasil ekstraksi objek pada citra

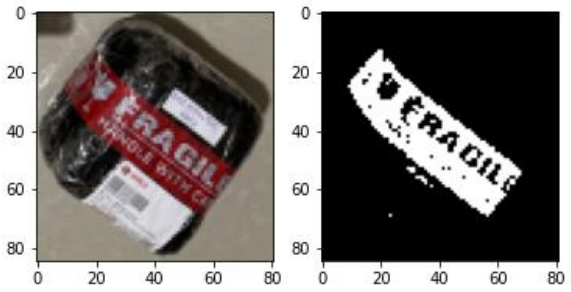
*G. Mengambangkan Citra Objek berdasarkan Warna*

Citra objek yang dihasilkan dari operasi ekstraksi objek akan dilakukan pengambangan pada citra objek berdasarkan warna untuk menentukan paket pengiriman rapuh atau tidak rapuh.

Pada OpenCV di Python, pengambangan pada citra berdasarkan warna dilakukan dengan memanggil fungsi, sebagai berikut.

```
cv.inRange(img, low_range, high_range)
```

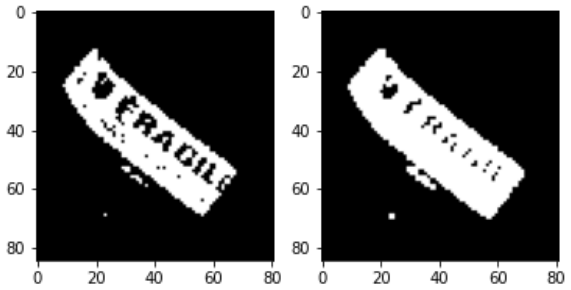
`img` merupakan citra masukan, `low_range` merupakan batas bawah warna dalam model warna HSV, dan `high_range` merupakan batas atas warna dalam model warna HSV. Pada sistem ini, batas bawah warna merah dalam model warna HSV bernilai [170, 50, 50] dan batas atas warna merah dalam model warna HSV bernilai [180, 255, 255].



**Gambar III.8** Perbandingan citra objek semula (kiri) dengan citra objek hasil pengambangan (kanan)

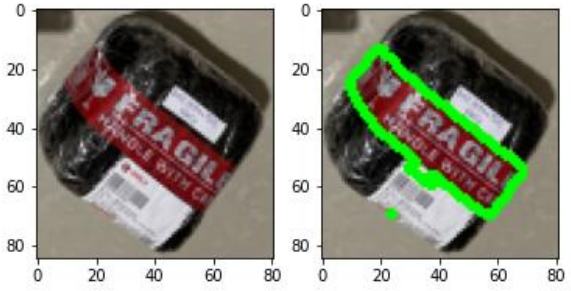
*H. Mendilatasi Citra Objek*

Citra objek yang telah dilakukan pengambangan akan dilakukan dilasi agar tepi citra menjadi lebih tebal seperti pada operasi dilatasi citra sebelumnya.



**Gambar III.9** Perbandingan citra objek masukan (kiri) dengan citra objek hasil *dilate* (kanan)

*I. Mencari Kontur Citra Objek*

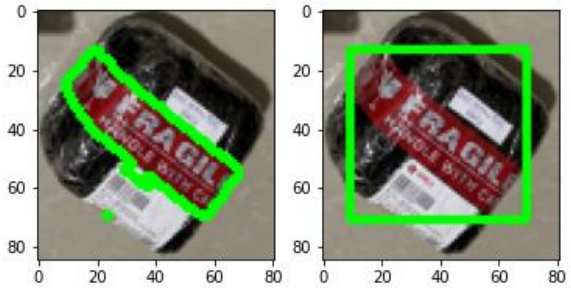


**Gambar III.10** Perbandingan citra objek semula (kiri) dengan citra objek hasil pencarian kontur (kanan)

Setelah dilasi dilakukan pada citra objek, operasi pencarian kontur dilakukan untuk menemukan tepi dari region label *fragile*. Operasi ini dilakukan sama seperti operasi pencarian kontur citra pada tahap sebelumnya.

*J. Menyaring Kontur Citra Objek*

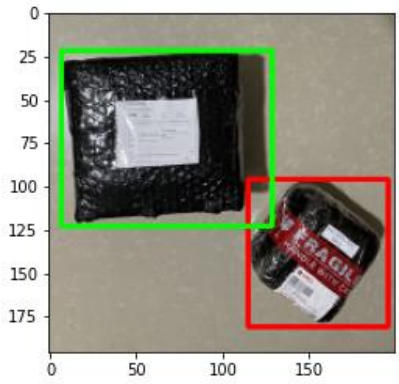
Penyaringan kontur citra objek dilakukan untuk membuang region yang tidak diharapkan. Seperti pada operasi penyaringan kontur citra sebelumnya, operasi ini menggunakan **`cv.boundingRect`** dengan cara dan nilai parameter yang sama. Namun, jika pada operasi sebelumnya region yang memiliki luas kurang dari 2000 pixel persegi dihilangkan, pada operasi ini regional yang kurang dari 10% dari luas citra objek akan dihilangkan.



**Gambar III.11** Perbandingan citra objek masukan (kiri) dengan citra objek hasil *boundingRect* dan penyaringan kontur (kanan)

*K. Melabeli Citra*

Pada tahap ini, objek-objek (pada kasus ini adalah paket pengiriman) pada citra telah diketahui apakah objek tersebut merupakan objek yang rapuh atau objek yang tidak rapuh. Pelabelan dilakukan dengan menggunakan **`cv.rectange`** pada OpenCV. Hijau menandakan objek tersebut tidak rapuh, sedangkan merah menandakan objek tersebut rapuh.



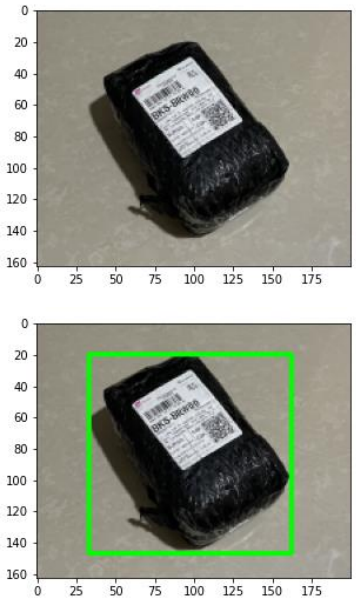
**Gambar III.12** Citra hasil pelabelan citra

## IV. Eksperimen dan Hasil Analisis

*A. Eksperimen*

Dalam menguji coba keandalan sistem yang dibangun, dilakukan eksperimen beberapa skenario citra masukan; yaitu citra yang berisi satu paket pengiriman, citra yang berisi dua paket pengiriman atau lebih, citra yang berisi dua paket pengiriman yang berhimpitan, citra yang berisi paket pengiriman berwarna merah, dan citra dengan pencahayaan dan/atau latar yang kurang baik. Berikut hasil eksperimen yang dilakukan dengan skenario citra masukan pada sistem pendeteksi paket pengiriman rapuh dengan segmentasi citra.

*1) Satu paket pengiriman*

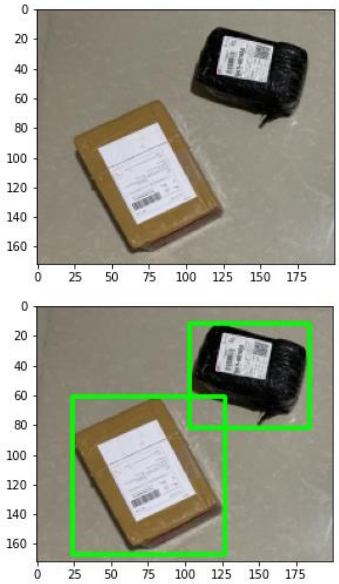


**Gambar IV.1** Perbandingan citra masukan (atas) dengan citra hasil deteksi (bawah) pada skenario satu paket pengiriman

Eksperimen ini dilakukan dengan menggunakan citra yang berisi 1 (satu) buah paket pengiriman. Jika dilihat pada Gambar IV.1, sistem dapat bekerja dengan sangat baik pada skenario ini. Selain itu, setiap operasi pada sistem juga memberikan citra keluaran yang sesuai dengan harapan.

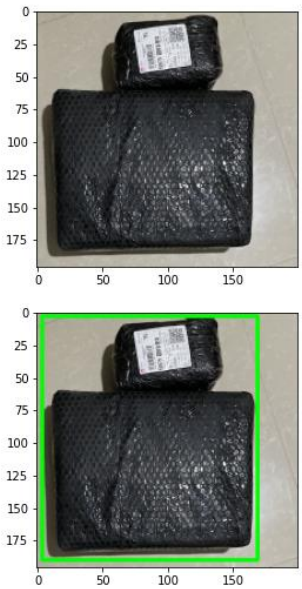
*2) Dua paket pengiriman atau lebih*

Eksperimen kedua ini dilakukan dengan menggunakan citra yang berisi 2 (dua) buah paket pengiriman. Jika dilihat pada Gambar IV.2, sistem juga dapat bekerja dengan sangat baik pada skenario ini. Sistem dapat dengan benar mendeteksi bahwa kedua paket pengiriman tersebut tidak rapuh, yang dibuktikan dengan hasil deteksi kedua paket pengiriman diberi label warna hijau.



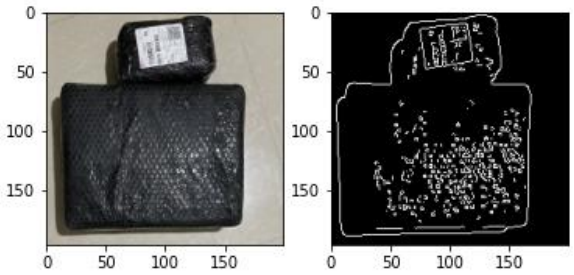
**Gambar IV.2** Perbandingan citra masukan (atas) dengan citra hasil deteksi (bawah) pada skenario dua paket pengiriman

*3) Dua paket pengiriman berhimpitan*



**Gambar IV.3** Perbandingan citra masukan (atas) dengan citra hasil deteksi (bawah) pada skenario dua paket pengiriman berhimpitan

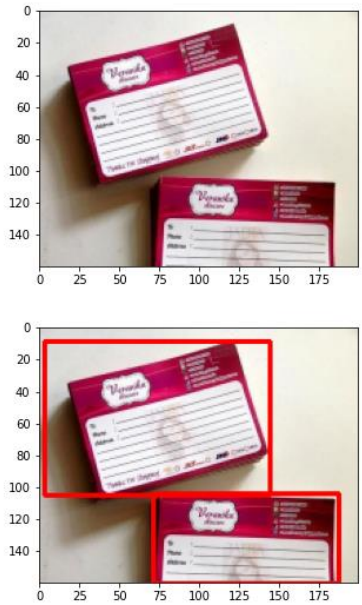
Eksperimen ini dilakukan dengan menggunakan citra yang berisi 2 (dua) buah paket pengiriman yang saling berhimpitan. Jika dilihat pada Gambar IV.3, sistem gagal dalam mendeteksi paket pengiriman ini. Sistem tidak mengetahui terdapat 2 (dua) buah paket pengiriman pada citra, melainkan paket menganggap hanya terdapat 1 (satu) buah paket pengiriman.



**Gambar IV.4** Perbandingan citra masukan (kiri) dengan citra hasil operator Canny (kanan) pada skenario dua paket pengiriman berhimpitan

Jika dilakukan *tracing* di setiap operasi pada sistem, citra keluaran pada operasi pendeteksian tepi dengan operator Canny yang menyebabkan sistem menganggap hanya terdapat 1 (satu) buah paket pengiriman. Pendeteksian tepi pada citra ini menghubungkan dua paket pengiriman menjadi satu, seperti yang dapat dilihat pada Gambar IV.4. Oleh karena itu, operasi selanjutnya menganggap citra masukan hanya memiliki 1 (satu) buah objek. Hal ini tentu masuk akal jika dilihat berdasarkan cara kerja pendeteksian tepi dengan operator Canny, namun penggunaan metode ini kurang tepat jika sistem akan menghadapi skenario seperti ini.

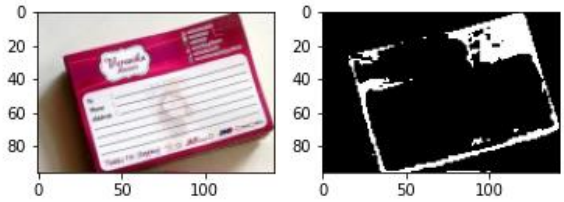
*4) Paket pengiriman berwarna merah*



**Gambar IV.5** Perbandingan citra masukan (atas) dengan citra hasil deteksi (bawah) pada skenario paket pengiriman berwarna merah

Eksperimen ini dilakukan dengan menggunakan citra yang berisi 2 (dua) buah paket pengiriman berwarna merah. Jika dilihat pada Gambar IV.5, sistem gagal dalam mendeteksi paket pengiriman ini. Sistem berhasil mendeteksi terdapat 2 (dua)

buah paket pengiriman pada citra, namun sistem gagal dalam mendeteksi paket pengiriman rapuh atau tidak rapuh pada citra. Sistem menganggap kedua paket pengiriman merupakan paket yang rapuh, namun pada kenyataannya paket pengiriman tersebut tidak rapuh (tidak terdapat label *fragile*).



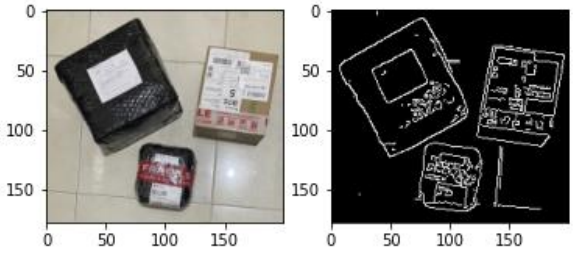
**Gambar IV.6** Perbandingan citra objek semula (kiri) dengan citra objek hasil pengambangan (kanan) pada skenario paket pengiriman berwarna merah

Jika dilihat berdasarkan cara kerja sistem, sistem berjalan dengan baik sesuai dengan harapan cara kerja sistem, karena komposisi warna merah yang melebihi 10% dari luas objek akan dilabeli *fragile*. Inilah yang menjadi pengecualian pada sistem. Sistem akan gagal dalam mendeteksi suatu paket pengiriman rapuh atau tidak rapuh jika terdapat komposisi warna merah sebesar 10% selain yang ditujukan untuk label *fragile*. Dalam menentukan suatu objek rapuh atau tidak rapuh, metode ini kurang memberikan hasil yang memuaskan jika sistem akan menghadapi skenario seperti ini.

*5) Pencahayaan dan/atau latar yang kurang baik*

**Gambar IV.7** Perbandingan citra masukan (atas) dengan citra hasil deteksi (bawah) pada skenario pencahayaan dan/atau latar yang kurang baik

Eksperimen ini dilakukan dengan menggunakan citra yang berisi 3 (dua) buah paket pengiriman dengan pencahayaan dan/atau latar yang kurang baik. Jika dilihat pada Gambar IV.7, sistem gagal dalam mendeteksi jumlah paket pengiriman yang ada pada citra. Sistem menganggap terdapat 4 (empat) paket pengiriman, yang mana pada kenyataannya hanya terdapat 3 (tiga) paket pengiriman. Naumn, sistem berhasil dalam mendeteksi paket pengiriman rapuh atau tidak rapuh.



**Gambar IV.8** Perbandingan citra objek semula (kiri) dengan citra objek hasil operator Canny (kanan) pada skenario pencahayaan dan/atau latar yang kurang baik

Jika dilakukan *tracing* pada sistem, operator Canny mendeteksi latar yang dekat dengan pantulan lampu, seperti yang dapat dilihat pada Gambar IV.8. Hal ini yang menyebabkan adanya 1 (satu) paket pengiriman tambahan yang terdeteksi oleh sistem. Oleh karena itu, sistem akan kesulitan dalam mendeteksi paket pengiriman rapuh jika sistem akan menghadapi citra dengan pencahayaan dan/atau latar yang kurang baik seperti pada skenario ini.

## *B. Hasil Analisis*

Berdasarkan hasil eksperimen, diperoleh banyak sekali batasan yang ada pada sistem pendeteksi paket pengiriman rapuh dengan pendeteksian tepi dan model warna. Sistem X berjalan dengan lebih baik jika sistem tidak menghadapi citra yang berisi 2 (dua) atau lebih paket pengiriman yang berhimpitan, sistem tidak menghadapi citra yang berisi paket pengiriman berwarna merah, dan sistem tidak menghadapi pencahayaan dan/atau latar yang kurang baik (atau rumit).

Dengan begitu, berikut beberapa faktor yang mempengaruhi akurasi sistem pendeteksi paket penfiriman rapuh dengan segmentasi citra.

- Sudut pandang kamera,
- Kerapatan antar paket pengiriman,
- Warna paket pengiriman,
- Pencahayaan, dan
- Latar (medan, atau *terrain*)

## V. KESIMPULAN DAN SARAN

Pendeteksian paket pengiriman rapuh dengan menggunakan metode pendeteksian tepi dan model waran mampu memberikan akurasi yang lebih baik jika posisi kamera perlu sejajar dengan latar, antar paket pengiriman diberikan ruang, warna paket tidak mengikuti warna *fragile* pada umumnya, pencahayaan yang tidak memberikan pantulan yang tajam, dan latar dengan pola

minimal atau polos. Namun kurang memberikan akurasi yang tinggi jika hal tersebut tidak terpenuhi.

Dengan lingkungan yang tidak seperti itu, metode ini baik digunakan pada tahap *preprocessing* yang kemudian akan ditindaklanjuti oleh metode yang lebih *advanced*, seperti sistem OCR untuk mendeteksi kata *fragile* atau metode lainnya.

## REFERENSI


[1] https://www.merdeka.com/uang/kemenkop-catat-transaksi-belanja-online-meningkat-26-persen-selama-pandemi.html, diakses pada tanggal 23-05-2021.

[2] https://inet.detik.com/cyberlife/d-5155740/masa-pandemi-transaksi-shopee-di-q2-2020-naik-hingga-130, diakses pada tanggal 23-05-2021.

[3] https://industri.kontan.co.id/news/di-tengah-pandemi-kinerja-jne-tumbuh-30, diakses pada tanggal 23-05-2021.

[4] https://www.google.com/search?q=paket+rusak, diakses pad tanggal 23-05-2021.

[5] Munir, R. 2021. Slide Pendeteksian Tepi. https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2020-2021/13-Pendeteksian-Tepi-2021.pdf

[6] Munir, R. 2021. Slide Warna. https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2020-2021/15-Warna-2021.pdf

[7] Munir, R. 2021. Slide Segmentasi Citra. https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2020-2021/17-Segmentasi-Citra-2021.pdf


## PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bekasi, 24 May 2021

13517034 Muhammad Fariz Luthfan Wakan