Implementasi Pengenalan Tulisan Cetak Menggunakan Tesseract Dan Komunikasi Jaringan Menggunakan Python

1st Muhammad Rizkiyanda
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
rizkiyanda@student.telkomuniversity.ac.id

2nd Wahmisari Priharti
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
wpriharti@telkomuniversity.ac.id

3rd Iswahyudi Hidayat
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
iswahyudihidayat@telkomuniversity.ac.id

***Abstrak* — Dokumen cetak masih digunakan di banyak industri untuk menyimpan informasi penting dalam satu format: faktur, kwitansi, dan dokumen cetak lainnya. Ini menjadi masalah ketika formulir informasi digital diperlukan dari dokumen cetak. Oleh karena itu diperlukan suatu sistem yang dapat mengubah gambar dokumen untuk dicetak menjadi string, sehingga tidak perlu memasukkan data secara manual ke dalam komputer. Saat ini, salah satu teknologi yang mengenali huruf dalam gambar adalah Optical Character Recognition (OCR) yang terdapat di dalamnya. diprogram untuk melakukan segmentasi, ekstraksi fitur, klasifikasi, pelatihan, dan pengenalan. Namun, akurasi Tesseract bergantung pada kualitas gambar, noise, dan ukuran font, sehingga diperlukan pemrosesan gambar tambahan. Oleh karena itu, alat pemindaian portabel untuk dokumen cetak dikembangkan dalam penelitian ini Menggunakan OCR Tesseract dengan langkah-langkah pemrosesan gambar: Grayscale, Unsharp Mask, Threshold, dan ekstensi dengan pustaka menggunakan komunikasi jaringan menggunaka python. Arial, *fake receipt* dan calibri dengan ukuran 11, 12 dan 14 memberikan tingkat akurasi rata -rata sebesar 93.33%.**

Kata kunci— Optical Characther Recognition*, *Tesseract,* Pemindaian Portbel*, Pemprosesan gambar, Komunikasi Jaringan menggunakan python, paramiko.

I. PENDAHULUAN

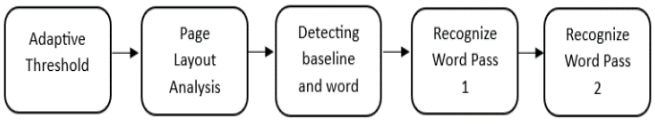
Di zaman modern ini, teknologi sudah menjadi bagian dari kehidupan sehari-hari masyarakat. Teknologi telah memungkinkan manusia untuk melakukan aktivitas sehari-hari. Seiring dengan pertumbuhan tersebut, aplikasi seluler juga berkembang pesat dan banyak diminati oleh berbagai kalangan karena kemudahan penggunaannya dan dapat digunakan di mana saja. Salah satu sistem operasi yang paling populer adalah Android, sekitar 74,85% pasar sistem operasi perangkat seluler didominasi oleh Android. [1]. Saat ini sudah ada teknologi pemindai (scanner), namun pemindai hanya mampu untuk menangkap citra dari dokumen cetak, kemudian citra tersebut perlu diubah menggunakan perangkat lunak tambahan jika ingin mendapatkan hasil pindaian dalam bentuk *data string* atau teks digital.

Proyek ini menggunakan algoritma pengenalan karakter optik untuk pengenalan karakter. Secara umum, tujuan OCR adalah untuk mengubah gambar teks yang diketik, ditulis tangan, atau dicetak menjadi teks [2]. Teknologi OCR mengklasifikasikan karakter-karakter atau pola-pola pada citra untuk dicocokkan dengan huruf atau angka sehingga menghasilkan data yang mampu diproses oleh komputer *(data string)* [3]. Salah satu cara untuk mengaplikasikan OCR adalah dengan menggunakan OCR *engine*. Tesseract merupakan salah satu perangkat lunak dari OCR *engine open source* dengan tingkat akurasi sebesar 96,38% untuk dokumen cetak [4]. Akurasi dari tesseract sangat berpengaruh kepada kualitas citra dan ukuran *font* dokumen cetak. Akurasi Tesseract turun drastis jika kualitas citra kurang dari 300 dpi.

Penelitian saat ini sedang merancang sebuah alat pemindai yang memiliki kemampuan untuk mengenali teks pada dokumen cetak. Tujuan dari penelitian ini adalah untuk mengkonversi teks pada dokumen cetak menjadi teks digital yang dapat diproses oleh komputer dengan tingkat akurasi lebih dari 90% (error 10%). Alat ini menggunakan teknologi OCR dengan metode Tesseract. Dokumen cetak ditangkap menggunakan kamera yang terintegrasi dengan Raspberry Pi 4. Sebelum citra dokumen dikenali menggunakan Tesseract, citra tersebut akan melalui tahapan pengolahan citra untuk meminimalisasi noise dan meningkatkan kualitas citra. Setelah proses pengenalan teks selesai, hasil yang diperoleh dari alat ini adalah data string yang berisi teks dari dokumen cetak. Untuk pengiriman data, kami merekomendasikan penggunaan paramiko sebagai jembatan untuk mengirimkan data yang sudah berbentuk string ke perangkat yang ditentukan. Penggunaan paramiko memungkinkan pengiriman data tanpa kabel, sehingga memudahkan perpindahan data antar perangkat tanpa perlu menggunakan kabel fisik.

II. KAJIAN TEORI

A. OCR Tesseract



GAMBAR 1
Arsitektur Tesseract

OCR adalah sistem yang sudah lama dikembangkan. Tahun 1914, Emanuel Goldberg telah mulai membuat sistem OCR yang berfungsi untuk telegram dan alat baca untuk

orang tunanetra. Sistem OCR terus dikembangkan hingga kini sehingga dapat menghasilkan akurasi yang lebih baik bahkan dalam situasi-situasi yang dimana karakter sulit untuk dikenali. Pengaplikasian OCR pada program memerlukan Library. Library memungkinkan algoritma pembaca dapat di-compile oleh program yang mengaplikasikan OCR tersebut [5]. Arsitektur Tesseract dijelaskan pada Gambar 1.

Proses pengenalan huruf pada citra diawali dengan adaptive thresholding. Ini diikuti oleh analisis tata letak atau analisis tata letak, di mana Tesseract mengidentifikasi antara area teks dan non-teks. Berikutnya adalah deteksi garis dan kata. Proses ini untuk mendeteksi teks miring. Terakhir, proses pengenalan karakter pada citra dilakukan dua kali. Proses kedua akan mendeteksi ulang karakter yang tidak dapat dikenali pada input pertama [5].

Gambar yang dihasilkan oleh kamera atau pemindai belum cukup siap untuk pengenalan huruf dengan OCR Tesseract karena gangguan gambar. Akibatnya, akurasi OCR Tesseract menurun atau mesin bahkan tidak dapat mengenali huruf pada gambar. Ini dapat dihindari dengan langkah-langkah pemrosesan gambar sebelum pengenalan huruf [6].

B. Luminance Grayscale

Teknik grayscale merupakan teknik pengolahan citra dengan mengubah warna yang muncul pada citra yaitu merah, hijau dan biru, menjadi hitam putih. Alasan utama penggunaan gambar skala abu-abu adalah untuk menyederhanakan algoritme sistem dan mengurangi beban komputasi. Gambar berwarna memerlukan pelatihan yang lebih banyak sehingga sistem tidak efisien [7]. Algoritma yang paling baik untuk diimplementasikan pada sistem OCR adalah algoritma Brightness [8]. Algoritma ini dijabarkan dalam persamaan (1) dan penerapannya pada citra ditunjukkan pada Gambar 2 .

Luminance adalah ukuran fotometrik intensitas cahaya per satuan luas cahaya yang merambat dalam arah tertentu, Y merupakan reprentasi dari luminan dari suatu image, komponen – komponennya adalah :

Gambar 2
Hasil citra pada proses grayscaling

$$Luminance = 0{,}299 \times \mathrm{R} + 0{,}587 \times G + 0{,}114 \times B \qquad (1)$$

dimana:
R = nilai warna merah
G = nilai warna hijau
B = nilai warna biru

C. Thresholding

Thesholding adalah proses mengekstraksi objek yang ada pada citra dengan menerapkan nilai threshold pada tiap piksel sehingga setiap piksel dapat diklasifikasikan sebagai foreground dan background. Nilai threshold memberikan nilai biner pada tiap piksel di mana jika nilai piksel biner lebih dari nilai threshold, maka piksel tersebut akan diberi nilai satu dan sebaliknya nol [7]. Metode thresholding umum digunakan dalam pengenalan teks dan simbol, contohnya untuk memproses dokumen [10].

Salah satu algoritma yang banyak digunakan untuk pengolahan citra untuk mengenali karakter adalah thresholding. Kelebihan dari thresholding adalah algoritmanya sendiri yang mencari nilai threshold untuk setiap citranya [10]. Hasil proses dari thresholding ditunjukkan pada Gambar 3. Di dunia Penginderaan Jauh, *thresholding* merupakan istilah yang sudah sangat familiar. Tidak ada praktisi Penginderaan Jauh yang tidak pernah mendengar kata thresholding. Thresholding atau proses penentuan ambang/batas nilai pixel pada citra digital.

D. Paramiko

Python sendiri memiliki beberapa jenis library yang digunakan untuk implementasi sistem otomasi jaringan diantaranya adalah Paramiko dan Netmiko. Namun kedua jenis library Python tersebut memiliki perbedaan ketika akan digunakan untuk proses otomasi jaringan. Library Paramiko dapat langsung menjalankan script yang berisi konfigurasi perangkat, kemudian dikirimkan ke perangkat yang akan dikonfigurasi menggunakan protokol SSH (Secure Shell).

Gambar 3
Hasil citra pada proses grayscaling

Proses pengumpulan data yang berupa citra dari setiap ukuran dan jenis font yang didapatkan dari tesseract. Data hasil konfigurasi akan dikirimkan ke perangkat melalui jaringan menggunakan protokol yang sama yaitu SSH. Pada penelitian yang dilakukan oleh peneliti sebelumnya, berhasil dibuat aplikasi dengan menggunakan Bahasa pemrograman Python dan library yang digunakan adalah Paramiko [11].

III. METODE PENELITIAN

Metode penelitian pada perancangan OCR Scanner pada penelitian ini antara lain:

A. Studi literatur

Tahapan ini dilakukan untuk memahami teori dan juga mencari informasi dalam pengerjaan penelitian melalui jurnal, website dan dosen yang terkait dalam penelitian ini sebagai pemecah masalah.

B. Pengumpulan data

Proses pengumpulan data yang berupa citra dari setiap ukuran dan jenis font yang didapatkan dari tesseract. Data set yang berada pada tesseract memiliki 400.000 baris teks yang mencakup 4500 dari jenis yang akan digunakan sebagai proses klasifikasi.

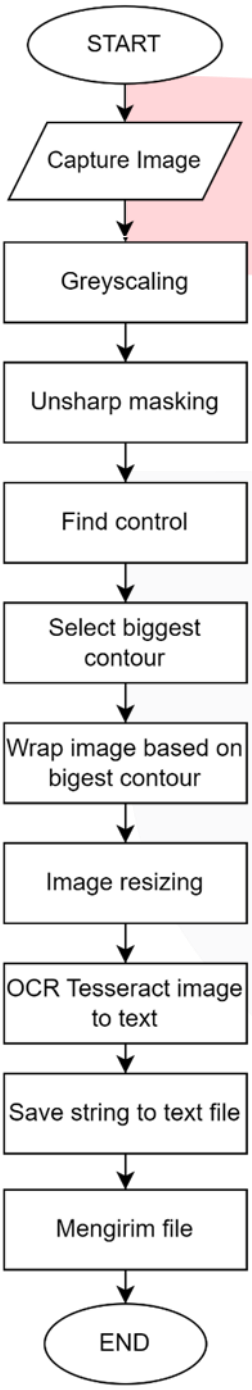
C. Pengelolahan Data

Pemrosesan data pada tesseract OCR melibatkan beberapa langkah, antara lain image preprocessing, text segmentation, character recognition, post-processing, dan ekstraksi data. Tesseract memiliki beragam pengaturan dan profil yang dapat disesuaikan untuk mengoptimalkan

performa OCR tergantung kebutuhan dan jenis gambar yang sedang diproses. Semakin baik kualitas gambar, semakin besar font gambar dan gambar pra-proses, semakin baik peluang untuk mendapatkan hasil OCR yang akurat.

D. Evaluasi dan Kesimpulan

Singkatnya, Tesseract OCR adalah teknologi yang cocok dan berguna untuk mengenali teks dari gambar atau dokumen dan kemudian mengubahnya menjadi string untuk transfer dokumen yang mudah, dapat menghemat waktu dan tenaga. Dengan preprocessing yang tepat dan evaluasi hasil yang cermat, Tesseract dapat memberikan hasil OCR yang akurat yang berguna untuk banyak tujuan pengolah kata. .



Gambar 4
Flowchart tesseract

IV. PERANCANGAN

Program ini menggunakan bahasa pemrograman Python dan jaringan komunikasi untuk mengirim file ke perangkat yang dituju. Pertama-tama, program akan menyalakan kamera dan mengambil gambar. Gambar tersebut kemudian diubah menjadi citra grayscale untuk memudahkan pengolahan karakter. Selanjutnya, citra grayscale diproses dengan teknik thresholding untuk memperjelas karakter pada dokumen. Setelah itu, program akan mencari kontur pada citra dan mencari kontur terbesar yang merupakan kontur kertas. Setelah koordinat kontur kertas ditemukan, program akan memotong citra sesuai dengan koordinat kontur tersebut.

Tahapan ini b Thresholding hasilkan citra yang hanya berisi karakt isi dan memperbaiki posisi dokumen jika saat pengambilan citra dokumen masih miring. Jika pengguna menekan tombol spasi, citra akan melewati tahap resizing untuk meningkatkan resolusi gambar saat dilakukan rekognisi karakter. Sebelum dilakukan tahap rekognisi, citra akan melalui proses dilatation untuk memperjelas karakter pada citra. Terakhir, hasil citra akan direkognisi menggunakan Tesseract dan hasilnya akan disimpan dalam file teks. Gambar 4 menunjukkan flowchart validasi alat dari program tersebut.

V. HASIL DAN PEMBAHASAN

Sistem pemindai portable menggunakan kamera digunakan untuk mengambil gambar seperti yang ditunjukkan pada Gambar 5. Selanjutnya dilakukan pengolahan citra dimana pada tahap pertama dilakukan Luminance grayscale seperti yang ditampilkan pada Gambar 6. Nilai sigmaX menentukan nilai simpangan baku dari nilainilai pada kernel.

Gambar 5 *Citra hasil tangkapan kamera*

Hasil dari citra luminance grayscaling ditampilkan pada Gambar 6. Pengolahan citra yang dilakukan setelah itu adalah thresholding seperti ditampilkan pada Gambar 6. Gambar ini merupakan citra yang kemudian diolah menggunakan Tesseract untuk diubah menjadi karakter atau teks digital.

Selain itu, rentang kesalahan yang tinggi pada pengujian font berukuran kecil juga dapat disebabkan oleh kemampuan kamera yang kurang mampu untuk menangkap citra secara detail dan kualitas citra yang kurang baik untuk pengenalan tulisan. Berdasarkan penelitian sebelumnya, akurasi OCR Tesseract akan maksimal jika kualitas citra lebih dari 300 dpi, namun citra yang dihasilkan oleh sistem ini hanya memiliki resolusi sebesar 75 dpi. Hal ini dapat disebabkan oleh beberapa faktor, seperti kualitas kamera yang digunakan atau ketidakmerataan pencahayaan saat pengambilan gambar dilakukan.

Kapabilitas sistem diuji melalui empat jenis pengujian, yaitu uji identifikasi kata, uji identifikasi kata dalam kalimat, uji identifikasi kata dalam paragraf, dan uji identifikasi kata dalam dokumen struk. Semua pengujian menggunakan jenis font Arial, Calibri, dan Fake Receipt dengan ukuran font 11, 12, dan 14. Jenis dan ukuran font ini dipilih karena sering digunakan dalam dokumen cetak, terutama kuitansi dan resi. Pada setiap dokumen, citra diambil sebanyak enam kali. Performansi sistem dihitung berdasarkan persentase kesalahan yang dihitung dengan persamaan:

$$Kesalahan\ (\%) = (100 - (\frac{B}{S} \times 100) \quad (2)$$

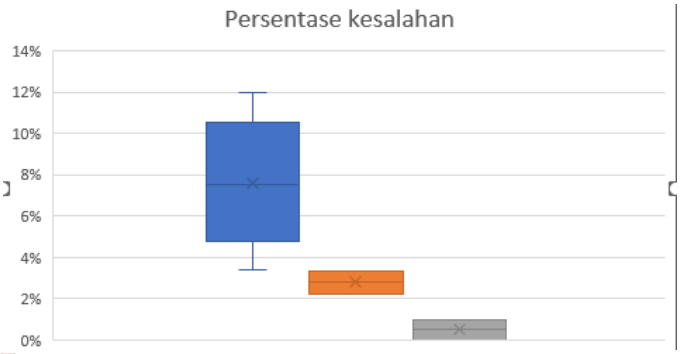
Dalam hal ini, variabel B adalah jumlah kata yang berhasil diidentifkasi alat dan S adalah total semua kata pada dokumen. Persentase kesalahan pada setiap pengujian ditunjukkan masing-masing pada Gambar 8.

Berdasarkan persentase kesalahan hasil pengujian yang ditunjukkan pada Gambar 8 dan 9, dapat disimpulkan bahwa sistem yang dihasilkan memiliki rentang kesalahan yang cukup tinggi (80-90%) untuk mengidentifikasi karakter-karakter yang memiliki font berukuran kecil. Hal ini terjadi karena peningkatan resolusi citra karakter tidak sebanding dengan peningkatan ukuran font. Pada citra karakter "B" dengan ukuran font 11, terlihat bahwa garis atas karakter tidak terlalu jelas sehingga pada beberapa pengujian, Tesseract mengenali huruf tersebut sebagai huruf "K". Terdapat perbedaan resolusi citra huruf pada masing-masing ukuran font 11, 12, dan 14 untuk font Arial.

Jika dianalisis berdasarkan jenis font dengan kondisi ukuran font yang sama (ukuran font 12), akurasi sistem tidak berbeda secara signifikan. Pada pengujian pengenalan huruf dalam kata, akurasi jenis font Arial (96,65%) untuk font *fake receipt* adalah (97,75%) dan untuk calibri (93,33%). Berdasarkan hasil ini, dapat dinilai bahwa akurasi sistem pengenalan karakter menggunakan Tesseract tidak bergantung kepada jenis font namun bergantung pada ukuran font yang digunakan.

VI. KESIMPULAN

Sistem pemindai dokumen cetak portabel menggunakan Raspberry Pi 4 dan Arducam 16MP IMX 519 menggunakan metode Tesseract berhasil mengidentifikasi tulisan pada dokumen cetak pada jenis font Arial, calibri dan *fake receipt* dengan akurasi terbaik untuk ukuran font 16. Sistem ini juga berhasil mengubah citra menjadi teks digital dengan rata-rata persentase kesesalahan 3,38% - 10,37% untuk font 11, 2,25% - 3,35% untuk font 12 dan 0% - 0,57% untuk font 14.



Gambar 8
Presentase kesalahan pengujian identifikasi pada tulisan cetak

REFERENSI

[1] Mobile Operating System Market Share Worldwide – April 2019. 2019. Statcounter. Retrieved from http://gs.statcounter.com/os-market-share/mobile/worldwide

[2] Kayethri D, Dharunya R, Harini M, "RECOGNIZATION HOLIC- MEDICINE DETECTION USING DEEP LEARNING TECHNIQUES" Computer Science & Engineering: An International Journal (CSEIJ), Vol 12, No 6, December 2022 DOI:10.5121/cseij.2022.12614 145

[3] A. Chaudhuri, K. Mandaviya, P. Badelia, and S. K. Ghosh, "Optical character recognition systems," in Studies in Fuzziness and Soft Computing, vol. 352, Springer Verlag, 2017, pp. 9–41. doi: 10.1007/978-3- 319-50252-6 2.

[4] Ariefsyahnakri "PERANCANGAN PEMINDAI DOKUMEN CETAK PORTABEL MENGGUNAKAN TESSERACT DAN OPENCV" Diterima pada 20 Februari 2022; diterbitkan pada 31 Desember 2022.

[5] Ali Firdaus, M. Syamsu Kurnia, Tia Shafera, Wahyu Istalama Firdaus. "Implementasi Optical Character Recognition (OCR) Pada Masa Pandemi Covid-19" J urnal JUPITER, Vol. 13, No. 2, Bulan Oktober, Tahun 2021 Hal. 188 – 194.

[6] N Venkata Rao and D. Ascssastry, "Optical Character Recognition Technique Algorithms," Journal of Theoretical and Applied Information Technology, vol. 20, no. 2, 2016, [Online]. Available: www.jatit.org

[7] E. H. A. and R. N, "OCR Accuracy Improvement on Document Images Through a Novel Pre-Processing Approach," Signal & Image Processing: An International Journal, vol. 6, no. 4, pp. 01–18, Aug. 2015, doi: 10.5121/sipij.2015.6401.

[8] C. Kanan and G. W. Cottrell, "Color-to-grayscale: Does the method matter in image recognition?," PLoS ONE, vol. 7, no. 1, Jan. 2012, doi: 10.1371/journal.pone.0029740.

[9] G. Deng, “A generalized unsharp masking algorithm,” IEEE Transactions on Image Processing, vol. 20, no. 5, pp. 1249–1261, May 2011, doi: 10.1109/TIP.2010.2092441.

[10] J. Yousefi, “Image Binarization using Otsu Thresholding Algorithm,” 2011, doi: 10.13140/RG.2.1.4758.9284.

[11]. R. A. Wiryawan and N. R. Rosyid, “Pengembangan Aplikasi Otomatisasi Administrasi Jaringan Berbasis Website Menggunakan Bahasa Pemrograman Python,” Simetris J. Tek. Mesin, Elektro dan Ilmu Komput., vol. 10, no. 2, pp. 741–752, 2019.