Pemanfaatan Teknik Image Processing untuk Menyelesaikan Permainan Sudoku

Giant Andreas Tambunan / 13519127
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): giantandreas0311@gmail.com

***Abstrak*—Sudoku merupakan permainan teka-teki angka yang biasa ditemukan sehari-hari. Makalah ini akan membahas pengimplementasian penyelesaian permainan sudoku secara otomatis dengan menggunakan *image processing* dan AI. Teknik *image processing* digunakan untuk ekstraksi informasi dari gambar *board* permainan sudoku. Teknik-teknik tersebut meliputi *edge detection*, image *segmentation*, dan pengenalan karakter. Sedangkan AI digunakan untuk menyelesaikan permainan sudoku berdasarkan informasi yang diekstraksi.**

Kata kunci—Sudoku; image segmentation; edge detection; kontur; character recognition;

I. Pendahuluan

Permainan *puzzle* merupakan salah satu jenis permainan yang sudah populer sejak beberapa dekade lalu. Permainan *puzzle* populer dikarenakan permainan ini yang dapat mengasah ketajaman otak maupun menguji kecerdasan. Permainan *puzzle* dibuat berdasarkan logika untuk hiburan namun tidak jarang juga memunculkan permasalahan matematika. Permainan *puzzle* memiliki banyak jenis.

Salah satu contoh permainan *puzzle* yang sangat populer adalah Sudoku. Hal tersebut ditandai dengan banyaknya majalah maupun koran yang mendedikasikan sebagian dari kolomnya untuk Sudoku.

Permainan Sudoku pertama kali diperkenalkan pada tahun 1895 di surat kabar Prancis yang dipengaruhi matematikawan Swiss Leonhard Euler sebagai Latin Square. Namun, permainan Sudoku modern yang sekarang kita kenal sekarang ditemukan oleh Howard Garns, seorang *puzzle inventor* dari Connersville, Indiana pada tahun 1979 yang dipublikasikan pada majalah *Dell Pencil Puzzles and Word Games*. Pada saat itu nama permainan tersebut dikenal sebagai *Number Place*, yang mana dinamakan berdasarkan permainan itu sendiri yang meletakkan angka-angka pada kotak 9x9.

Permainan tersebut pertama kali dikenalkan di Jepang pada tahun 1984 dimana nama permainan tersebut diubah menjadi Sudoku. Nama Sudoku merupakan singkatan dari frasa bahasa Jepang “Sūji wa dokushin ni kagiru” yang berarti sebuah angka hanya boleh muncul satu kali. Sudoku menjadi sangat populer di Jepang oleh karena ketidakcocokan bahasa Jepang dengan *crossword puzzle*, sehingga *puzzle* angka seperti Sudoku lebih populer.

Sudoku merupakan permainan *puzzle* yang cukup terkenal untuk semua kalangan usia dan bisa diakses dengan berbagai cara seperti koran, majalah, aplikasi, web, dan lain-lain. Pada semua platform tersebut, Sudoku yang disediakan disajikan dalam bentuk gambar board yang umumnya memiliki *grid* 9x9.

Makalah ini akan membahas pemanfaatan dan pengimplementasian teknik-teknik yang ada pada *image processing* untuk membantu menyelesaikan permainan *puzzle* Sudoku. Bentuk solusi yang ditawarkan akan dibagi menjadi tiga tahapan utama, yaitu ekstraksi informasi dari gambar *board* Sudoku, penyelesaikan Sudoku dengan menggunakan AI, dan penggambaran hasil solusi permainan Sudoku.

II. Dasar Teori

A. Sudoku

Sudoku merupakan sebuah *puzzle* yang memiliki n buah kolom dan n buah baris dengan n adalah integer [1]. Terdapat beberapa variasi permainan Sudoku yang bergantung pada seberapa besar ukuran boardnya, yaitu 4x4, 5x5, 9x9, 16x16, dan 25x25. Ukuran *board* permainan Sudoku yang paling biasa ditemui dan yang paling standar adalah 9x9. Makalah ini akan membahas permainan Sudoku dengan ukuran 9x9.

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Fig. 1. Contoh Gambar Permainan Sudoku

Pada permainan Sudoku, terdapat beberapa angka secara random akan diisikan yang kemudian akan dilengkapi oleh pemain dengan beberapa aturan tertentu. *Grid* pada setiap permainan Sudoku terdiri nxn kotak. Dan setiap kotak tersebut akan dikelompokkan menjadi sebuah kelompok yang disebut minigrid. Aturan pengelompokan tersebut berbeda-beda bergantung kepada ukuran *grid* Sudoku. Pada permainan Sudoku dengan *grid* 9x9, setiap 3x3 kotak akan dikelompokkan menjadi sebuah *minigrid*. Dengan begitu, Sudoku dengan ukuran *grid* 9x9 akan memiliki 9 *minigrid*.

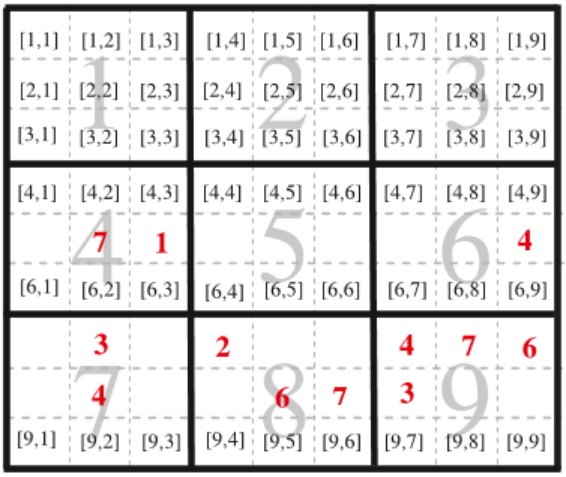


Fig. 2. Permainan Sudoku dengan *grid* 9x9 yang memiliki 9 *minigrid* berukuran 3x3

Tujuan permainan Sudoku adalah melengkapi semua kotak kosong yang ada dengan aturan sebagai berikut.

- Untuk setiap baris pada board tidak ada kotak yang memiliki angka yang sama
- Untuk setiap kolom pada board tidak ada kotak yang memiliki angka yang sama
- Untuk setiap *minigrid* yang ada pada board tidak ada kotak yang memiliki angka yang sama.

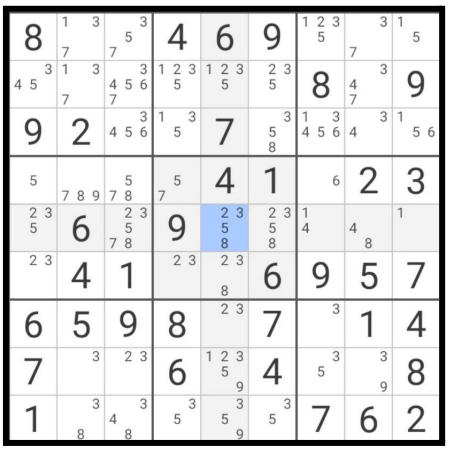


Fig. 3. Permainan Sudoku dengan semua kotak kosong yang sudah diisikan dengan semua kemungkinan angka yang memenuhi aturan

B. *Edge Detection (Pendeteksian Tepi)*

Edge atau tepi merupakan perubahan nilai intensitas keabuan yang mendadak (besar) dalam jarak yang singkat [2]. Tepi memiliki arah, dan arah ini berbeda-beda bergantung pada perubahan intensitas. Tepi biasanya terdapat pada batas antara dua daerah yang berbeda intensitas dengan perubahan yang sangat cepat di dalam citra. Terdapat 4 macam tepi yaitu: tepi curam (step edge), tepi landai (ramp edge), tepi garis (line edge), dan tepi atap (roof edge).

Edge detection atau pendeteksian tepi adalah sebuah proses yang bertujuan untuk meningkatkan penampakan garis batas atau objek di dalam citra. Pendeteksian tepi mengekstraksi representasi gambar garis-garis di dalam citra. Pendeteksian tepi juga dapat berguna dalam mengenali objek di dalam citra (image recognition). Pendeteksian citra dapat dipahami dengan pendekatan diferensial oleh karena perubahan intensitas yang besar dalam jarak yang singkat dipandang sebagai fungsi yang memiliki kemiringan yang besar. Kemiringan fungsi dapat dihitung dengan turunan pertama (gradient).

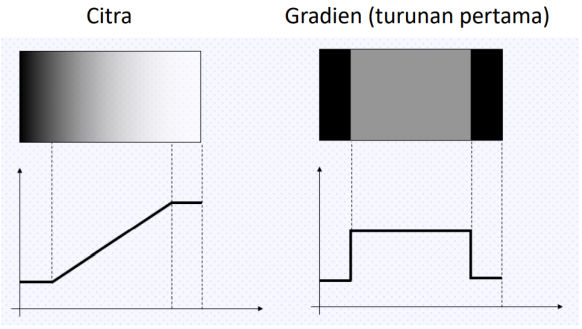


Fig. 4. *Edge detection* dengan pendekatan turunan pertama

Selain dengan pendekatan turunan, *edge detection* juga dapat dilakukan dengan beberapa pendekatan lain dengan operator yang berbeda. Salah satu operator tersebut adalah operator *Laplace* yang menggunakan operasi turunan kedua. Operator ini dapat mendeteksi tepi secara lebih akurat dibanding dengan menggunakan pendekatan turunan pertama. Hal tersebut disebabkan Operator ini menggunakan persilangan nol pada turunan kedua untuk menentukan apakah sebuah pixel merupakan *edge*/tepi atau tidak. Jika turunan kedua berubah dari negatif menjadi positif ataupun sebaliknya maka titik dimana turunan kedua melewati angka nol akan dianggap sebagai edge. Operasi turunan kedua dapat dilakukan pada suatu citra dengan mask konvolusi pada persamaan berikut.

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Biarpun dapat mendeteksi *edge* dengan lebih baik, masih terdapat beberapa kelemahan pada operator *Laplace*. Kelemahan tersebut terletak pada pekanya operator tersebut pada derau sehingga kerap kali muncul tepi-tepi palsu oleh karena pengaruh derau.Untuk mengatasi kekurangan tersebut

maka citra perlu ditapis terlebih dahulu untuk menghilangkan derau. Salah satu penapis yang baik digunakan untuk kasus tersebut adalah penapis *Gaussian*. Untuk memaksimalkan efektivitas pendeteksian tepi, matriks kernel penapis dapat digabungkan dengan matriks operator yang kemudian menciptakan operator baru yaitu *Laplace of Gaussian* atau biasa disingkat LoG. Berikut merupakan matriks kernel 5x5 untuk deteksi tepi dengan operator LoG.

$$\begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}$$

Selain dengan menggunakan pendekatan turunan pertama dan turunan kedua, terdapat 3 operator lain yang menggunakan *mask* konvolusi untuk mendapatkan citra *edge*. Ketiga operator tersebut merupakan operator Sobel, operator Roberts, dan operator Prewitt.

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{dan} \quad S_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

(a)

$$P_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{dan} \quad P_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

(b)

$$R_+ = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad \text{dan} \quad R_- = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

(c)

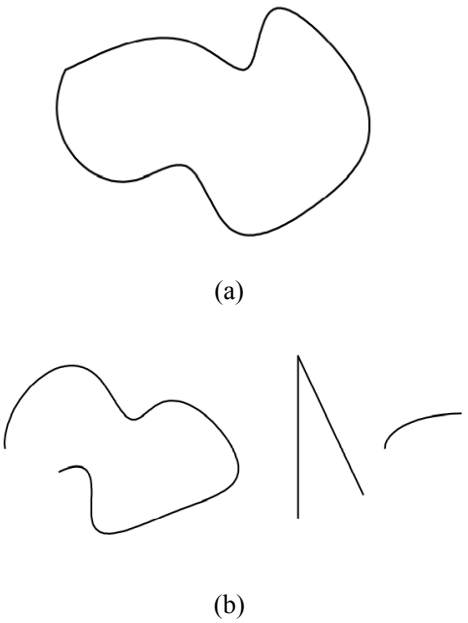
Fig. 5. (a) *mask* konvolusi operator Sobel, (b) *mask* konvolusi operator Prewitt, dan (c) *mask* konvolusi operator Roberts

Selain operator-operator tersebut terdapat operator Canny. Operator Canny merupakan operator yang paling populer karena dapat mendeteksi tepi dengan lebar tepat 1 *pixel*. Yang membedakan operator Canny dengan operator-operator lain adalah penggunaan dua nilai pengambang (*threshold*), T1 dan T2, dimaan T2 lebih besar daripada T1 yang memungkinkan untuk mendeteksi dua jenis tepi, yaitu tepi kuat (*strong edges*) dan tepi lemah (*weak edges*). Apabila magnitudo pixel hasil perhitungan pada ruang citra gradien memiliki nilai yang lebih besar dari pengambang T2, maka pixel tersebut termasuk tepi kuat. Untuk semua pixel yang terhubung dengan tepi kuat dan memiliki nilai magnitudo yang lebih besar dari pengambang T1, maka pixel tersebut termasuk tepi lemah.

C. Kontur

Pendeteksian tepi menghasilkan citra biner (hitam putih) dimana pixel-pixel tepi didefinisikan sebagai warna putih (bernilai 1) sedangkan pixel-pixel bukan tepi didefinisikan sebagai warna hitam (bernilai 0). Akan tetapi, sebuah citra hasil *edge detection* belum dapat memberikan informasi yang berarti. Oleh karena itu, citra tersebut masih perlu diolah lebih lanjut sehingga dapat mengenali bentuk-bentuk tertentu yang sederhana misalnya garis lurus, kotak, lingkaran, elips, dan sebagainya.

Kontur merupakan rangkaian pixel-pixel yang membentuk suatu batas region (*region boundary*) [6]. Kontur memiliki dua jenis, yaitu kontur terbuka dan kontur tertutup. Kontur tertutup berkoresponden pada suatu batas yang mengelilingi sebuah region. Pixel-pixel pada suatu daerah yang tertutup dapat dideteksi dengan algoritma pengisian (*filling algorithm*). Batas daerah digunakan untuk mendeskripsikan bentuk objek pada tahap analisis citra. Sedangkan kontur terbuka merupakan fragmen garis ataupun bagian dari batas daerah yang tidak membentuk sebuah sirkuit.



(a) (b)

Fig. 6. (a) Kontur tertutup (b) Kontur tertutup

D. Image Segmentation (Segmentasi Citra)

Image Segmentation atau Segmentasi Citra termasuk tahapan pada analisis citra. Tahapan ini dilakukan setelah ekstraksi fitur, sebelum akhirnya akan dilanjutkan dengan tahapan klasifikasi. *Image Segmentation* adalah teknik pada pemrosesan citra yang digunakan untuk membedakan daerah-daerah pada citra. Tujuan dari segmentasi citra adalah untuk merepresentasikan citra dengan lebih sederhana sehingga lebih mudah untuk dianalisis. Segmentasi citra memisah misahkan setiap pixel pada gambar pada kelompok-kelompok tertentu sehingga setiap segmen/region yang dihasilkan memiliki sifat yang mirip.

Hasil dari *Image Segmentation* adalah sejumlah segmen atau region dimana setiap pixel pada citra merupakan anggota dari segmen atau region tersebut. Pengelompokan tersebut dilakukan berdasarkan parameter yang bermacam-macam misalnya warna, tekstur, maupun intensitas sedemikian sehingga setiap pixel pada region yang sama memiliki karakteristik yang mirip berdasarkan parameter tersebut. Selain itu, pixel pada region tertentu juga harus memiliki perbedaan yang signifikan dengan pixel yang berada pada region lain. Oleh karena itu, Segmentasi Citra dapat dilakukan dengan analisis kontur dengan menggunakan teknik *edge detection*. Pendekatan ini disebut juga dengan metode Diskontinuitas.

Segmentasi merupakan tahapan yang penting pada *image analysis* karena memudahkan untuk menganalisis region yang penting pada citra saja. Pengaplikasian Segmentasi Citra banyak digunakan sehari-hari, misalnya dalam bidang medis (pendeteksian tumor, diagnosis anatomi, dan operasi), *autonomous vehicle* (membedakan antara mobil, jalan, pejalan kaki, langit, dsb), *objek recognition* (pengenalan objek pada gambar), *scene understanding*, dan lain sebagainya.

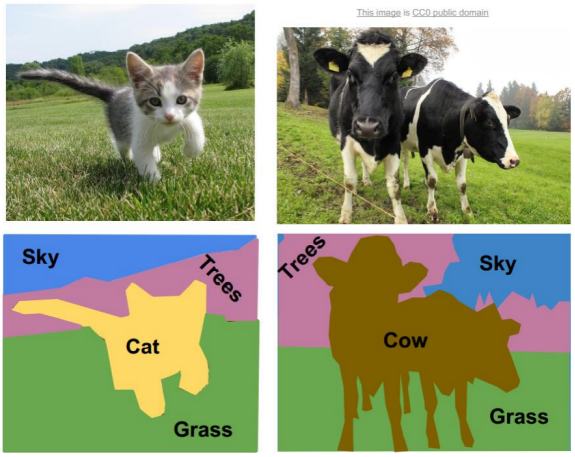


Fig. 7. Contoh pemanfaatan *Image Segmentation* pada *object recognition*

E. *py-sudoku*

py-sudoku merupakan *library* bahasa pemrograman Python yang digunakan untuk membuat maupun menyelesaikan mxn *puzzle* Sudoku. Beberapa fungsionalitas *py-sudoku* adalah membuat *basic puzzle* Sudoku, menyelesaikan sebuah model Sudoku, menampilkan papan Sudoku, mengimport papan Sudoku berdasarkan matriks input, mendeteksi apakah sebuah *puzzle* Sudoku tidak valid ataupun tidak memiliki solusi, dan menginisiasi *puzzle* Sudoku berdasarkan *seed* tertentu.

III. Implementasi Solusi

Solusi diimplementasikan dengan menggunakan bahasa pemrograman *MATLAB* versi *R2022a* untuk mengolah citra menggunakan teknik-teknik image processing dan *Python* versi 3.10 untuk menyelesaikan puzzle Sudoku. Solusi yang diimplementasikan terbatas pada Sudoku standar berukuran 9x9. Untuk puzzle Sudoku yang memiliki kemungkinan solusi lebih dari satu, maka program hanya akan mengembalikan salah satu solusi yang memungkinkan saja.

Solusi dapat dibagi menjadi tiga tahapan utama, yaitu ekstraksi informasi dari gambar input, penyelesaian puzzle Sudoku, dan penggambaran citra solusi puzzle Sudoku.

A. *Ekstraksi Informasi*

Input program berupa citra gambar permainan Sudoku yang belum diselesaikan. Citra masukan yang diinput mengandung board Sudoku yang tepat berbentuk persegi, sehingga program terbatas pada citra input yang terotasi maupun terdistorsi.

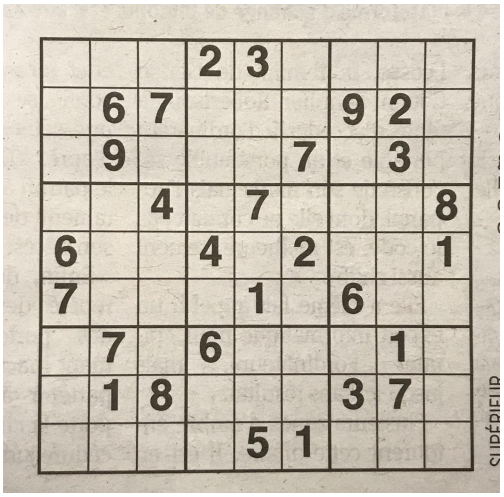


Fig. 8. Contoh citra input

Citra input kemudian akan melalui proses segmentasi untuk mengenali *board puzzle* Sudoku pada gambar. Segmentasi dilakukan dengan menggunakan metode Diskontinuitas dengan memanfaatkan hasil operasi *edge detection* pada citra input. Proses ini akan mendeteksi kotak dengan cara mendeteksi garis vertikal dan garis horizontal yang membentuk kotak. Kotak yang diduga sebagai *board* Sudoku akan diekstraksi untuk diproses kemudian.

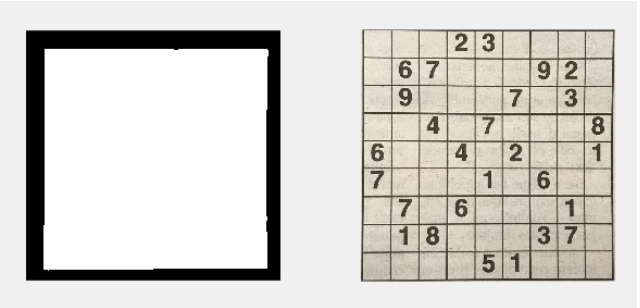


Fig. 9. Citra hasil segmentasi

Kemudian setiap angka pada kotak kotak yang ada pada *board* Sudoku hasil segmentasi akan diekstraksi. Sebelum setiap angka tersebut dapat diekstraksi, maka *board* tersebut harus tersebut dibagi menjadi citra-citra yang kecil yang merepresentasikan ke-81 kotak yang ada. Pembagian tersebut dilakukan dengan melakukan resizing pada image menjadi 900 x 900 pixel, kemudian melakukan *looping* sederhana untuk setiap 100 pixel secara vertikal dan secara horizontal. Proses

tersebut akan menghasilkan 81 citra yang berukuran 100x100 pixel.

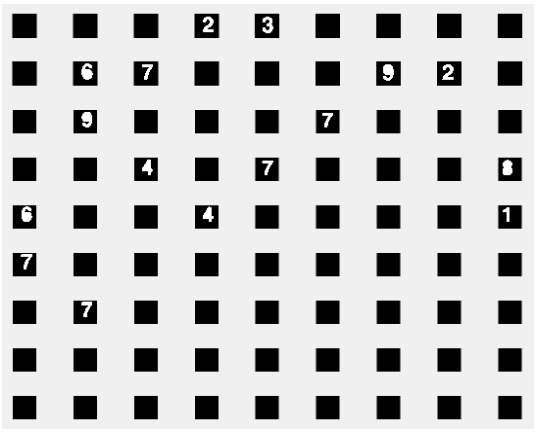


Fig. 10. Hasil pembagian kotak pada board Sudoku hasil segmentasi

Untuk masing-masing kotak yang dihasilkan pada pembagian tersebut, akan dilakukan pengenalan karakter berbasis korelasi gambar. Sebelumnya citra biner yang berperan sebagai *template* untuk karakter numerik 1-9 sudah disediakan. Kemudian untuk masing-masing kotak citra diatas akan dibandingkan dengan semua citra *template* untuk mendapatkan nilai korelasinya dimana label gambar dengan nilai korelasi tertinggi akan dipilih sebagai nilai yang terbaca. Sedangkan untuk kotak kosong akan dilabelis secara otomatis dengan nilai 0. Proses ini akan menghasilkan matriks 9 x 9 yang berisikan semua nilai numerik hasil pembacaan *board* Sudoku.

Matriks tersebut kemudian akan dikirimkan pada tahapan selanjutnya yaitu penyelesaian *puzzle* Sudoku dengan program Python.

B. Penyelesaian Puzzle

Matriks yang diterima dari tahapan sebelumnya merupakan matriks representasi dari citra input. Matriks tersebut kemudian akan diselesaikan pada tahapan ini dengan menggunakan bantuan *py-sudoku* oleh program Python. Program akan menginisiasikan model Sudoku dengan menggunakan matriks hasil pembacaan. Program kemudian akan melanjutkan dengan menyelesaikan *puzzle* yang menghasilkan matriks yang merepresentasikan solusi *puzzle* Sudoku yang sudah diinisiasi. Matriks tersebut kemudian dikirimkan kembali kepada program MATLAB untuk melanjutkan ke tahapan selanjutnya.

C. Penggambaran Citra Solusi

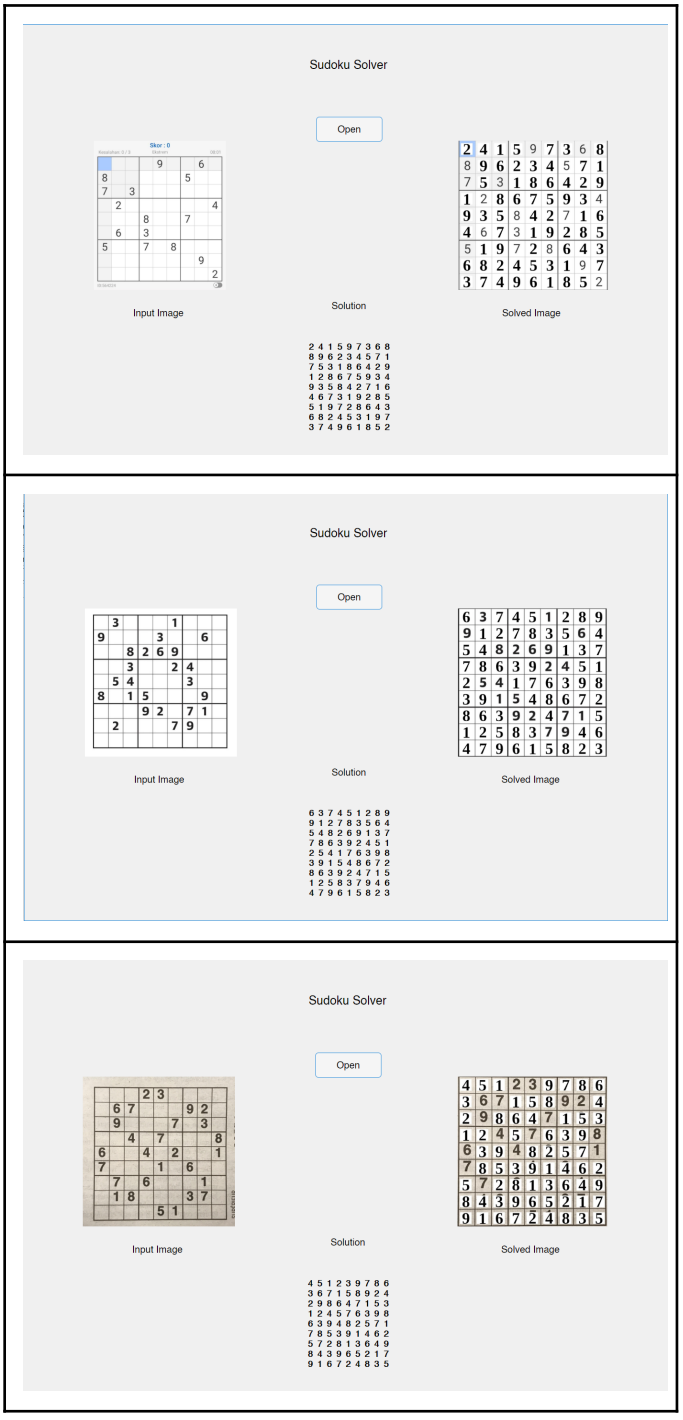
Pada tahapan ini, citra hasil segmentasi, matriks hasil pembacaan, dan matriks solusi dibutuhkan sebagai input. Hasil ataupun output dari tahapan ini berupa citra input yang sudah sudah diselesaikan.

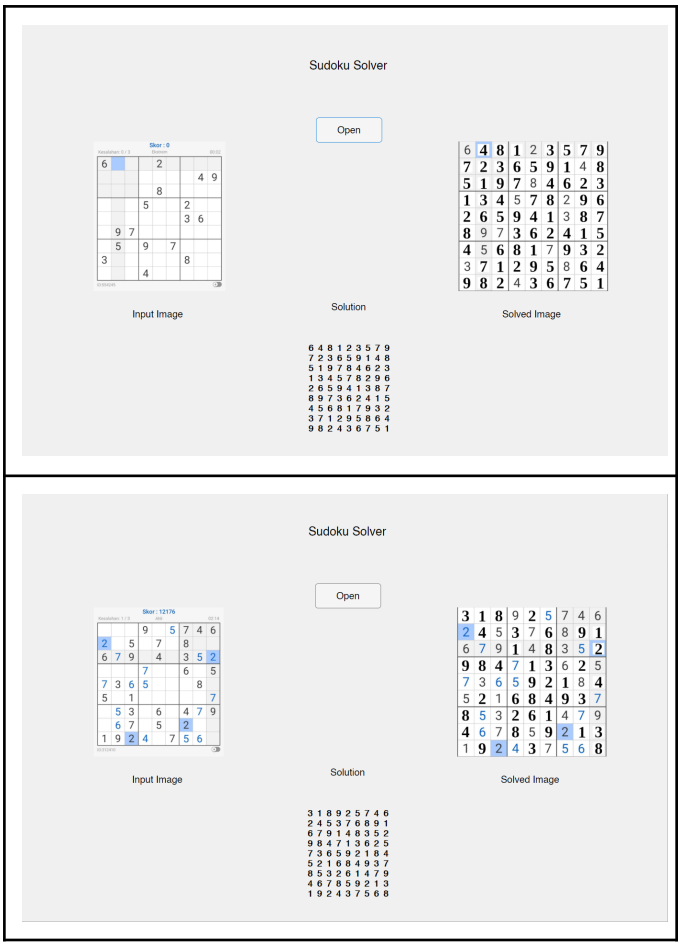
Untuk setiap kotak pada citra hasil segmentasi akan ditempeli angka solusi berdasarkan matriks input dan matriks solusi. Setiap kotak kosong berkoresponden dengan nilai 0 pada matriks input, dengan begitu semua kotak kosong dapat dideteksi dengan matriks input. Semua kotak kosong kemudian akan ditempeli dengan angka solusi yang didapatkan dari matriks solusi. Hasil akhir berupa citra *puzzle* Sudoku yang sudah diselesaikan yang dapat dianggap sebagai hasil akhir dari implementasi solusi ini.

IV. Pengujian

Berikut ini merupakan beberapa contoh citra input dan output hasil dari pengujian program yang dilakukan. Hasil pengujian akan disajikan dalam bentuk *screenshot* GUI yang telah disediakan.

TABLE I. Tabel Hasil Pengujian





V. Kesimpulan Dan Saran

Banyak hal yang dapat dilakukan untuk memudahkan kegiatan sehari-hari dengan memanfaatkan teknik-teknik image processing. Salah satunya adalah untuk menyelesaikan permainan Sudoku dengan lebih mudah. Dari hasil pengujian, tujuan solusi yang ditawarkan oleh makalah ini telah tercapai, yaitu untuk dapat menyelesaikan *puzzle* Sudoku dengan lebih mudah dan cepat.

Namun masih terdapat kelemahan pada beberapa bagian dari solusi. Kelemahan tersebut, terutama terletak pada pengenalan karakter pada bagian ekstraksi informasi yang masih menggunakan metode korelasi citra biner. Hal tersebut menimbulkan permasalahan kotak yang sebenarnya berisi angka gagal untuk dibaca. Penulis menyarankan penggunaan CNN untuk pengenalan karakter pada kotak-kotak Sudoku untuk akurasi yang lebih tinggi.

Pranala Terkait

Berikut ini merupakan link menuju *repository* dari solusi yang diimplementasikan: giantandreas/Sudolve (github.com)

Ucapan Terimakasih

Penulis mengucapkan puji dan syukur kepada Tuhan Yang Maha Esa atas berkat dan rahmat-Nya sehingga makalah berjudul “Pemanfaatan Teknik Image Processing untuk Menyelesaikan Permainan Sudoku” dapat diselesaikan dengan baik. Penulis juga mengucapkan terima kasih kepada Bapak Dr. Rinaldi Munir, S.T, M.T. selaku dosen pengampu kelas IF4073 Interpretasi dan Pengolahan Citra Semester ganjil tahun ajaran 2022/2023 yang telah banyak memberikan bimbingan dan ilmu terkait pemrosesan citra yang banyak digunakan dalam menyusun makalah ini. Ucapan terima kasih juga diucapkan kepada segenap keluarga dan juga teman-teman yang selalu mendukung dan memberikan semangat kepada penulis dalam menyusun makalah ini.

Referensi

[1] A. Kumar Maji, dan R. Kumar Pal, Sudoku Solver using Minigrid based Backtracking, IEEE International Advance Computing Conference (IACC), 2014

[2] R. Munir, “18 - Pendeteksian Tepi (edge detection) (bagian 1),” 2022.

[3] R. Munir, “18 - Pendeteksian Tepi (edge detection) (bagian 2),” 2022.

[4] H. Intelm, How to Solve Every Sudoku Puzzle, Vol.2, Geostar Publishing LLC. 2005.

[5] R. Munir, “22 - Segmentasi Citra (bagian 1),” 2022.

[6] R. Munir, “21 - Kontur,” 2022.

[7] Y.Ramadevi, T.Sridevi, B.Poornima, dan B.Kalyani, Segmentation And Object Recognition Using Edge Detection Techniques, International Journal of Computer Science & Information Technology (IJCSIT), Vol 2, No 6, December 2010

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.

Bandung, 17 Desember 2022

Giant Andreas Tambunan / 13519127