Aplikasi Pengenal Papan Reversi

Isabella Handayani Sumantri 13519081
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
135519081@std.stei.itb.ac.id

***Abstract*—Reversi merupakan permainan papan strategi yang marak dimainkan oleh segala kalangan umur. Pada makalah ini, akan dijelaskan detail pemecahan persoalan pengenalan papan permainan Reversi menggunakan teknik-teknik *image processing*. Hasil pengenalan akan dimanfaatkan untuk memainkan kembali permainan atau menganalisis hasil permainan.**

Keywords—Reversi; Image Processing; Pengenalan

I. Pendahuluan

Permainan Reversi merupakan permainan yang digemari oleh banyak kalangan umur. Reversi adalah sebuah permainan strategi yang dimainkan di papan berukuran 8x8. Permainan ini dimainkan oleh dua pemain yang masing-masing bergantian untuk menaruh *disk* ke papan, Pemain dengan *disk* paling banyak pada papan akan menang.

Permainan Reversi yang dimainkan secara fisik sering sekali tidak dapat diselesaikan. Pada makalah ini, akan dijelaskan detail pengenalan papan Reversi fisik untuk diubah ke bentuk digital. Perubahan ke bentuk digital akan memungkinkan pemain melanjutkan kembali permainannya. Selain itu, perubahan ke bentuk digital memungkinkan dimanfaatkannya hasil pengenalan sebagai bahan analisis dan berlatih. Pengenalan papan Reversi akan memanfaatkan beberapa teknik *image processing.*

II. Dasar Teori

A. *Reversi*

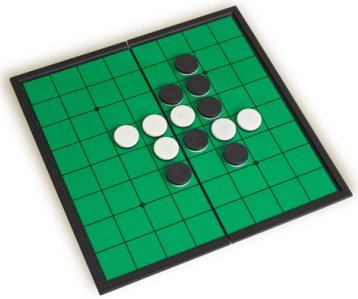

Fig. 1. Papan Permainan Reversi (Sumbe: Walmart)

Permainan Reversi adalah permainan strategi dua pemain pada sebuah papan 8x8. Permainan dilakukan dengan potongan bernama *disk*. *Disk* memiliki dua sisi berwarna hitam dan putih. Tujuan dari permainan ini adalah memiliki *disk* terbanyak pada papan sesuai dengan warna pemain. Saat permainan, pemain berusaha meraih poin terbanyak dengan membalikkan warna *disk* lawan menjadi warna *disk* sendiri.

Permainan Reversi memiliki beberapa aturan berikut:

a. Permainan dilakukan di papan berukuran 8x8.
b. Di awal permainan, isi papan kosong kecuali empat buah *disk* yang diletakkan di tengah. Terdapat dua buah *disk* berwarna putih dan dua buah *disc* berwarna hitam.
c. Pemain bergantian menaruh *disc* mereka di papan.
d. Pada setiap giliran, pemain harus menaruh *disk* mereka sehingga bisa mengelilingi dan menangkap (membalikkan) *disk* lawan. Sebuah *disk* dapat dibalik jika dikelilingi di kedua sisi.
e. Seorang pemain harus menangkap *disk* lawan jika memungkinkan. Jika tidak, giliran akan diskip.
f. Permainan akan berakhir jika papan penuh atau kedua pemain tidak dapat bergerak.
g. Pemain dengan *disk* paling banyak pada papan akan menang.
h. Jika jumlah *disk* kedua pemain sama, permainan berakhir seri.

B. *Warna*

Citra berwarna memiliki lebih banyak informasi dibandingkan dengan citra *grayscale* atau citra biner. Warna dapat digunakan sebagai alat identifikasi objek dan ekstraksinya. Dalam pemrosesan citra, model warna dapat digunakan sebagai sebuah standar untuk mengidentifikasi warna secara konsisten. Model warna yang sering digunakan dalam pengolahan citra antara lain, RGB, CMY/CMYK, YCbCr, HSI, HSV, dan XYZ.

Salah satu warna yang kerap digunakan pada pemrosesan citra adalah model warna HSV. Model warna HSV terdiri atas komponen *hue, saturation, value*. *Hue* adalah jenis warna sebenarnya pada warna dengan rentang nilai 0 sampai 2π, Komponen *saturation* adalah kemurnian warna dengan rentang nilai 0 sampai 1. Terakhir, komponen *value* adalah kecerahan dari sebuah warna dengan rentang nilai 0 dan 1. Model warna ini sering digunakan untuk melakukan segmentasi citra. Hal ini disebabkan informasi kromatik terpisah dengan informasi luminansi. Oleh karena itu,

memungkinkan identifikasi warna suatu objek tanpa terpengaruh oleh kecerahan atau intensitas warna.

C. *Citra Biner*

Citra biner merupakan citra yang memiliki dua nilai *graylevel* yaitu, nol yang melambangkan nilai putih dan satu yang melambangkan nilai hitam. Berikut adalah kegunaan dari citra biner:

1. Merepresentasikan citra hasil *edge detection*, untuk melakukan segmentasi objek.
2. Memisahkan objek dengan latar belakangnya.
3. Memfokuskan pada analisis bentuk morfologi
4. Menampilkan citra pada alat dengan resolusi intensitas satu bit

D. *Segmentasi Citra*

Segmentasi citra adalah proses pembagian citra menjadi sekumpulan *pixel* yang terhubung satu sama lain. Segmentasi dapat dilakukan untuk membagi citra menjadi region, struktur linier, atau bentuk 2D. Segmentasi citra bertujuan untuk membagi citra menjadi objek yang terpisah satu sama lain. Selain itu, dapat digunakan untuk memisahkan objek dari latar belakang. Kedua tujuan ini akan memudahkan analisis terhadap citra. Metode segmentasi citra umumnya dibagi menjadi dua pendekatan yaitu, diskontinuitas dan *similarity*.

Segmentasi citra melalui pendekatan diskontinuitas dapat memanfaatkan metode *edge detection*. Segmentasi dilakukan berdasarkan *pixel* tepi yaitu, *pixel* yang diidentifikasi dari perubahan cepat dalam nilai intensitas. Metode-metode *edge detection,* seperti metode berbasis gradien (Sobel, Prewitt, Roberts, Laplacian, LoG).

Segmentasi citra dengan pendekatan *similarity* ditentukan berdasarkan kemiripan area. Salah satu metode segmentasi berbasis *similarity* adalah *thresholding.* Metode pengambangan atau *thresholding* didasarkan pada nilai intensitas *pixel* dan nilai ambang *T*. Hasil segmentasi berupa sebuah citra biner. Teknik pengambangan dapat dibagi menjadi:

a. *Global thresholding*
   Nilai ambang bergantung pada semua nilai *pixel*.
b. *Local thresholding*
   Nilai ambang hanya bergantung pada *pixel* yang memiliki hubungan ketetanggaan.
c. *Adaptive thresholding*
   Nilai ambang berubah secara dinamis bergantung pada perubahan pencahayaan.

E. *Kontur*

Kontur adalah rangkaian *pixel* tepi yang membentuk batas daerah. Sebuah kontur dapat terbuka atau tertutup. Kontur tertutup adalah kontur yang berkoresponden dengan *region*. *Pixel-pixel* di daerah tertutup dapat diidentifikasi dengan menggunakan algoritma pengisian (*filling algorithm*) Batas daerah yang ditemukan dapat digunakan untuk mengidentifikasi sebuah objek.

F. *Image Warping*

Image warping adalah teknik untuk mengubah bentuk citra dengan mengubah koordinat *pixel* citra. *Image warping* dapat dilakukan menggunakan parameter-parameter matematika untuk mengubah bentuk atau struktur dari citra. Teknik ini disebut sebagai *parametric warp*. Parameter yang digunakan dapat berupa koordinat, skala, atau bentuk transformasi geometri lain. Beberapa contoh teknik *parametric warp* adalah:

1. *Affine*
   Transformasi *affine* adalah kombinasi dari transformasi linear dan translasi.
2. Translasi
   Translasi adalah proses pemindahan atau pergeseran sebuah citra atau objek dalam citra ke posisi yang berbeda. Hal ini dilakukan dengan mengubah koordinat *pixel* dalam citra.
3. Rotasi
   Rotasi adalah proses memutar sebuah citra ke arah yang berbeda. Rotasi digunakan untuk mengoreksi citra yang terbalik, mengubah orientasi citra, atau menyesuaikan posisi citra.
4. Perspektif
   Melakukan transformasi citra untuk membenarkan distorsi yang diakibatkan oleh sudut citra diambil.

G. *Clustering*

Clustering adalah sebuah teknik pembelajaran mesin yang melakukan pengelompokan titik data ke dalam *cluster* berdasarkan kesamaan. Tujuan dari *clustering* adalah menemukan pola dan hubungan dalam data. Salah satu algoritma yang umum digunakan untuk *clustering* adalah algoritma KMeans. Algoritma KMeans akan membagi data ke dalam sejumlah *cluster* yang ditentukan berdasarkan nilai rata-rata dari titik yang ada di dalam *cluster.*

H. *Similarity*

Nilai kemiripan dari dua hal dapat dihitung dengan pendekatan *euclidean distance*. *Euclidean distance* dapat didefinisikan dengan rumus berikut

$$d(p, q)^2 = (q_1 - p_1)^2 + (q_2 - p_2)^2$$

III. METODOLOGI

Dalam pembangunan aplikasi pengenal papan reversi, berikut adalah langkah-langkah yang akan dilakukan:

1. Melakukan konversi warna citra ke model warna HSV. Model warna HSV lebih umum digunakan karena identifikasi warna pada model HSV dapat dilakukan tanpa dipengaruhi *luminance*.
2. Melakukan segmentasi citra berdasarkan warna. Segmentasi dilakukan dengan *thresholding*. Akan ditentukan nilai ambang T_1 dan T_2. Kedua nilai

ambang digunakan untuk mengambil bagian warna hijau pada papan. Bagian berwarna hijau merupakan bagian yang ingin dianalisis.
3. Mencari kontur dari citra biner hasil segmentasi.
4. Menormalisasi hasil kontur.
5. Melakukan *image warping* berdasarkan perspektif.
6. Melakukan *clustering* pada citra yang sudah di-*warp* untuk memperoleh *centroid* warna.
7. Membagi citra ke 8x8 bagian terpisah.
8. Melakukan prediksi warna di tiap *square.*
9. Memasukkan hasil prediksi ke program permainan Reversi.

IV. Hasil dan Pembahasan

Aplikasi akan menerima masukan berupa papan permainan Reversi. Papan tersebut tidak berada di posisi yang mudah untuk mendeteksi *disk* pada *square*. Oleh karena itu, akan dilakukan *image warping* papan Reversi

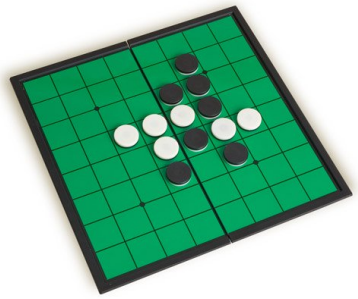

Fig. 2. Citra masukan program (Sumber: Walmart)

Sebelum melakukan *warping*, perlu dilakukan deteksi dari kontur papan Reversi. Citra dikonversi ke model warna HSV terlebih dahulu. Berikut adalah kode program dari konversi warna ke model warna HSV.

```
im_process = cv.cvtColor(image, cv.COLOR_BGR2HSV)
```

Hasil konversi ditunjukkan pada Fig 3.

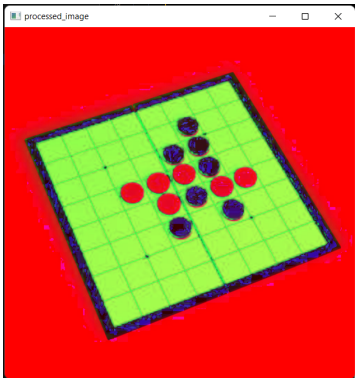

Fig. 3. Citra konversi ke model warna HSV (Sumber: Dokumentasi Pribadi)

Setelah itu, akan dilakukan segmentasi berdasarkan warna. Citra akan di-*threshold* untuk menghasilkan citra biner segmentasi. *Thresholding* dilakukan dengan menggunakan *upper bound* dan *lower bound* untuk warna hijau. Citra yang sudah di *threshold* akan melalui transformasi morfologi berupa *erosion* dan *dilation*. Transformasi ini dilakukan menggunakan metode morphologyEx. Kode program dari rangkaian pemrosesan tersebut didefinisikan pada potongan kode berikut,

```
def get_mask(self, image):
    lowerb = (43, 69, 42)
    upperb = (87, 255, 255)
    im_process = cv.inRange(image, lowerb, upperb)

    kernel =
cv.getStructuringElement(cv.MORPH_ELLIPSE, (10, 10))

    return cv.morphologyEx(im_process, cv.MORPH_CLOSE,
kernel)
```

Hasil dari pemrosesan tersebut dapat ditemukan pada Fig 4.

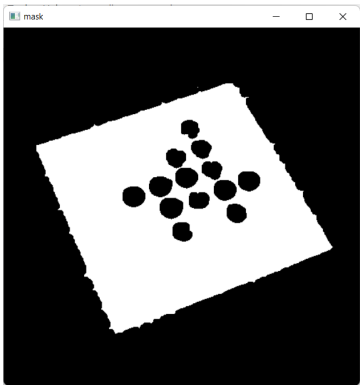

Fig. 4. Citra biner hasil *thresholding* (Sumber: Dokumentasi Pribadi)

Citra biner akan digunakan untuk mencari kontur dari papan. Pencarian kontur menggunakan fungsi findContours milik OpenCV. Fungsi ini mengimplementasikan algoritma [1]. Berikut adalah potongan kode pencarian kontur.

```
def find_contours(self, image):
    contours, _ = cv.findContours(image, cv.RETR_TREE,
cv.CHAIN_APPROX_SIMPLE)
    _, cnt = max([(cv.contourArea(i), i) for i in
contours], key=lambda x: x[0])
    return cnt
```

Hasil dari pencarian kontur dapat ditemukan pada Fig 5.

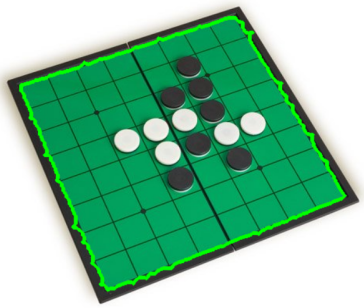

Fig. 5. Citra dengan kontur (Sumber: Dokumentasi Pribadi)

Kontur yang ditemukan kemudian dinormalisasi menggunakan fungsi approxPolyDp milik openCV. Fungsi ini mengimplementasikan algoritma Ramer-Douglas-Peucker. Tujuan dari algoritma tersebut adalah untuk mencari kurva yang mirip dengan masukan dengan menggunakan lebih sedikit titik. Berikut adalah kode program yang mengimplementasikan normalisasi kontur,

```
def get_board_contour(self, cnt):
    eps = 0.1 * cv.arcLength(cnt, True)
    approx = cv.approxPolyDP(cnt, eps, True)
    return approx
```

Hasil dari normalisasi kontur dapat ditemukan pada Fig 6.

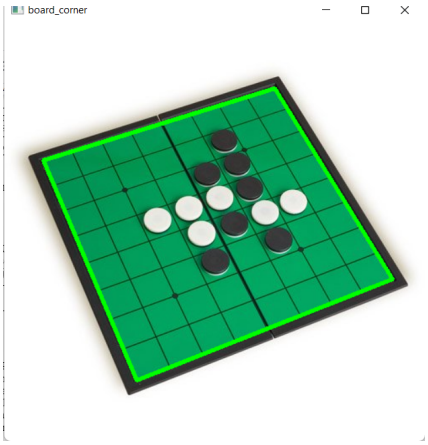

Fig. 6. Citra dengan kontur hasil normalisasi (Sumber: Dokumentasi Pribadi)

Hasil kontur yang sudah dinormalisasi kemudian digunakan untuk melakukan *warping* citra. *Image warping* dilakukan dengan memetakan titik ujung yang ditemukan pada kontur sebagai titik asal ke titik tujuan. Titik tujuan diatur menjadi (0, 0), (0, 500), (500, 0), dan (500, 500). Akan diperoleh citra hasil *image warping* berdasarkan perspektif. Berikut adalah kode program dari *warping* citra,

```
def transform(self, approx, image):
    src = np.float32(approx)
    dest = np.float32(
            [[0, 0], [0, self.size], [self.size,
self.size], [self.size, 0]]
        )

    M = cv.getPerspectiveTransform(src, dest)
    return cv.warpPerspective(image, M, (self.size,
self.size))
```

Hasil dari *warping* citra dapat ditemukan pada Fig 7.

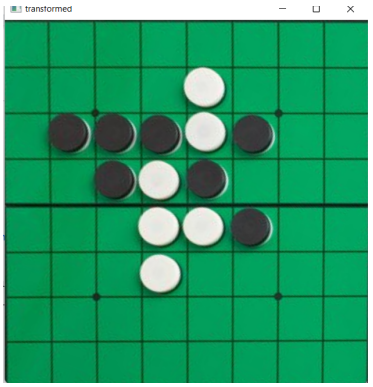

Fig. 7. Citra hasil *warping* (Sumber: Dokumentasi Pribadi)

Setelah melakukan *warping* terhadap citra, akan dilakukan deteksi warna setiap *disk* yang terdapat papan. Papan dibagi menjadi 64 bagian yang berukuran sama.

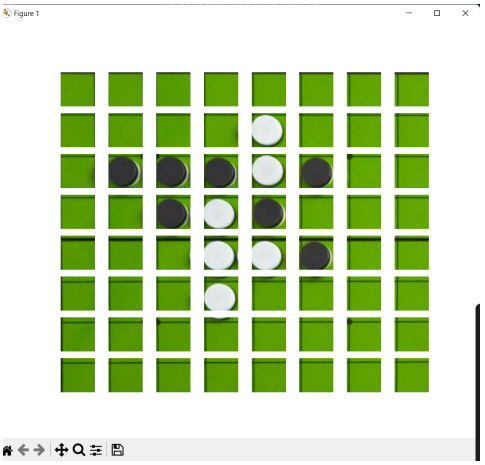

Fig. 8. Citra hasil *image warping* terbagi 64 bagian (Sumber: Dokumentasi Pribadi)

Untuk mendeteksi warna *disk* yang terdapat di papan, akan digunakan *clustering* menggunakan KMeans. KMeans menerima *input* berupa gambar yang belum dibagi menjadi 64 bagian. Nilai K dibuat tiga untuk mendeteksi nilai hijau, hitam, dan putih. *Centroid* hasil *clustering* akan menandakan nilai warna yang terdapat di citra.

```
def clusters(self, image):
    image = cv.cvtColor(image, cv.COLOR_BGR2RGB)
    self.clt = KMeans(n_clusters=3)
    self.clt.fit(image.reshape(-1, 3))

def get_piece(self, image):
    im_processed = cv.cvtColor(image,
cv.COLOR_BGR2RGB)

    pred = self.clt.predict(im_processed.reshape(-1,
3))

    return get_color_name(

self.clt.cluster_centers_[Counter(pred).most_common(1)
[0][0]]
    )
```

Untuk mengklasifikasikan nilai warna dari sebuah *disk*, akan dihitung nilai *similarity* dengan warna putih, hitam, dan hijau. Warna yang paling mirip dengan *centroid* akan menjadi hasil prediksi warna *disk*. Perhitungan similaritas ditunjukkan oleh kode program berikut.

```
KNOWN_COLORS = {"WHITE": (255, 255, 255), "BLACK": (0,
0, 0), "EMPTY": (50, 205, 50)}


def color_difference(color1, color2):
    return sum([math.sqrt((c1 - c2) ** 2) for c1, c2
in zip(color1, color2)])


def get_color_name(color):
    differences = [
        color_difference(color, known_color) for
known_color in KNOWN_COLORS.values()
    ]

    return
list(KNOWN_COLORS)[differences.index(min(differences))
]
```

Hasil prediksi papan ditunjukkan oleh senarai berikut.

```
[
        ['EMPTY', 'EMPTY', 'EMPTY', 'EMPTY', 'EMPTY',
        'EMPTY', 'EMPTY', 'EMPTY'],
        ['EMPTY', 'EMPTY', 'EMPTY', 'EMPTY', 'WHITE',
        'EMPTY', 'EMPTY', 'EMPTY'],
        ['EMPTY', 'BLACK', 'BLACK', 'BLACK', 'WHITE',
        'BLACK', 'EMPTY', 'EMPTY'],
        ['EMPTY', 'EMPTY', 'BLACK', 'WHITE', 'BLACK',
        'EMPTY', 'EMPTY', 'EMPTY'],
        ['EMPTY', 'EMPTY', 'EMPTY', 'WHITE', 'WHITE',
        'BLACK', 'EMPTY', 'EMPTY'],
        ['EMPTY', 'EMPTY', 'EMPTY', 'WHITE', 'EMPTY',
        'EMPTY', 'EMPTY', 'EMPTY'],
        ['EMPTY', 'EMPTY', 'EMPTY', 'EMPTY', 'EMPTY',
        'EMPTY', 'EMPTY', 'EMPTY'],
        ['EMPTY', 'EMPTY', 'EMPTY', 'EMPTY', 'EMPTY',
        'EMPTY', 'EMPTY', 'EMPTY']
]
```

Hasil prediksi akan dimasukkan ke program permainan Reversi agar dapat dimainkan kembali.

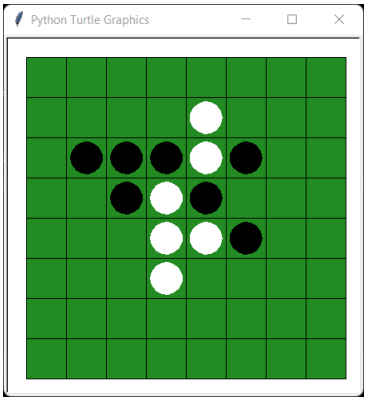

Fig. 9. Citra biner hasil *thresholding*

V. KESIMPULAN

Pada makalah ini, persoalan pengenalan Papan Reversi ke bentuk digital telah berhasil diselesaikan menggunakan rangkaian teknik *image processing*. Diperoleh bahwa langkah-langkah yang digunakan mampu mengenali *disk* papan Reversi dengan baik.

VIDEO LINK AT YOUTUBE

Untuk memberikan memberikan gambarkan tentang persoalan yang diangkat dan solusinya terlampir pranala video penjelasan berikut,

https://youtu.be/9vzdNSfLxmw

ACKNOWLEDGMENT

Penulis ingin memanjatkan rasa syukur kepada Tuhan yang Maha Esa atas rahmat penyertaan-Nya sehingga makalah ini

dapat diselesaikan. Penulis juga ingin mengucapkan terima kasih yang sebesar-besarnya atas pengajaran yang diberikan oleh Dr. Ir. Rinaldi, M.T. selaku dosen mata kuliah IF4073 Interpretasi dan Pengolahan Citra.

References

[1] Satoshi Suzuki and others. Topological structural analysis of digitized binary images by border following. Computer Vision, Graphics, and Image Processing, 30(1):32–46, 1985.

[2] Munir, Rinaldi. Citra Biner. https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2021-2022/20-Citra-Biner-2021.pdf. Diakses tanggal 18 Desember 2022, pukul 18.17 WIB

[3] Munir, Rinaldi. Kontur. https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2021-2022/21-Kontur-2021.pdf. Diakses tanggal 18 Desember 2022, pukul 19.01 WIB

[4] Munir, Rinaldi. Image Warping and Image Morphing. https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2022-2023/27-Image-Warping-Morphing-2021.pdf Diakses tanggal 18 Desember 2022, pukul 19.01 WIB. Diakses tanggal 18 Desember 2022, pukul 20.49 WIB

[5] Munir, Rinaldi. Warna. https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2021-2022/16-Warna-bagian1-2022.pdf Diakses tanggal 18 Desember 2022, pukul 19.01 WIB. Diakses tanggal 18 Desember 2022, pukul 21.29 WIB

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, 19 Desember 2022

Isabella Handayani Sumantri 13519081