Deteksi Teks Secara Otomatis Pada *Natural Image* Berbasis *Superpixel* Menggunakan *Maximally Stable Extremal Regions* dan *Stroke Width Transform*

Yohannes[#1]

[#]*Teknik Informatika, STMIK Global Informatika MDP*
Jalan Rajawali No. 14 Palembang
[1]yohannesmasterous@mdp.ac.id

Abstract — **Text detection in natural image is something to do before performing character recognition. The process of text detection plays an important role in the acquisition of information in an image. This research aims to detect text automatically in natural image based on superpixels with Maximally Stable Extremal Regions (MSER) and Stroke Width Transform (SWT). The superpixel method used is Simple Linear Iterative Clustering (SLIC). The SLIC method is used for segmenting text images into superpixel spaces. Image segmentation to superpixel aims to group pixels into homogeneous regions that capture redundant images. SLIC is a technique that effectively divides images into homogeneous regions (superpixels). Furthermore MSER is used as a feature to locate the text candidate region in a segmented image with superpixel. Then edge detection is done to validate the text area that has been found. Next, the SWT method is used to distinguish both text and non-text image regions. The dataset used is ICDAR 2003. Based on test result, MSER with superpixel is able to detect region of text in natural image. SWT is also able to recover the region which is the candidate of the text in natural image.**

Keywords— **Text detection, Superpixel, Simple Linear Iterative Clustering, Maximally Stable Extremal Regions, Stroke Width Transform**

I. PENDAHULUAN

Banyak sekali objek yang dijumpai mengandung unsur teks. Tidak hanya buku, namun juga objek seperti lukisan, papan pengumuman, rambu lalu lintas, logo, pakaian, alat-alat elektronik, dan lain-lain juga mengandung unsur teks. Teks merupakan informasi yang penting bagi siapapun.

Deteksi teks dalam sebuah citra merupakan hal utama sebelum melakukan pengenalan karakter. Deteksi teks pada sebuah *natural image* merupakan penelitian yang menarik [1]. Kesulitan yang dijumpai dalam deteksi teks adalah orientasi teks [2], warna teks yang hampir sama dengan warna latar atau objek lain [2], [3], *noise*, *blur* [4], [5], dan pencahayaan [5].

Natural image mempunyai tantangan tersendiri karena citra tersebut memiliki *background* yang kompleks. *Natural image* mempunyai beragam objek dalam sebuah gambar pemandangan sehingga wilayah teks yang akan dideteksi pada gambar tersebut menjadi tantangan bagi para peneliti. Tidak sedikit penelitian yang menerapkan berbagai metode untuk mendeteksi teks pada sebuah citra, diantaranya menggunakan teknik *clustering* berdasarkan gradien dan warna [2], [3], [6], teknik mendeteksi kesimetrian teks [7], dan mendeteksi lebar *stroke* pada wilayah kandidat teks [8], [5], [4]. Secara umum, ada dua tahapan untuk melakukan deteksi teks pada *natural image* dimulai dari penerapan metode untuk mendapatkan wilayah mana yang mengandung kandidat teks pada *natural image*. Setelah mendapatkan wilayah kandidat teks pada *natural image*, diperlukan metode untuk menentukan wilayah mana yang merupakan teks dan mana yang bukan teks pada kandidat yang telah didapatkan sebelumnya.

Metode yang telah dilakukan untuk mendapatkan wilayah yang mengandung kandidat teks meliputi *adaptive hierarchical clustering* [6], *maximally stable extremal regions* [4], [5], *edge detection* [8], *color reduction based extraction* [5], *visual saliency* [5], dan *gradient* and *color local feature* [2], [3]. Sedangkan metode yang telah dilakukan untuk menentukan wilayah mana yang merupakan teks dan mana yang bukan teks meliputi *single-link clustering* berbasis morfologi, *divisive hierarchical clustering* berbasis orientasi, *divisive clustering* berdasarkan proyeksi [6], *gradient vector flow* [7], dan *stroke width transform* [9], [8], [4].

Superpixel telah banyak digunakan dalam berbagai aplikasi *computer vision*. *Superpixel* mengelompokkan *pixel* ke dalam daerah homogen yang berisi informasi semantik dari bagian manusia dan objek. Selain itu, *superpixel* dapat menangkap redundansi citra dan memberikan fitur primitif sederhana yang kemudian diolah menjadi fitur citra lokal [10]. Representasi *superpixel* juga memungkinkan untuk dapat mendeskripsikan citra ke tingkat yang lebih tinggi dibandingkan dengan representasi *pixel*. *Superpixel* banyak digunakan untuk segmentasi citra.

Superpixel juga digunakan untuk *change detection* (deteksi perubahan wilayah). Pada penelitian [11],

superpixel digunakan untuk segmentasi citra *remote sensing* pada wilayah yang sama pada waktu yang berbeda. Hasil deteksi perubahan wilayah menggunakan *superpixel* dianggap lebih baik dibandingkan menggunakan *pixel* saja.

Di sisi lain, *superpixel* juga diterapkan di bidang *human action recognition* dalam hal meningkatkan akurasi pengenalan [12]. Berdasarkan hasil penelitian [12], pengenalan berbasis *superpixel* lebih baik dibandingkan dengan pengenalan tanpa berbasis *superpixel*. Di bidang *human interaction*, *superpixel* juga mampu meningkatkan akurasi [13]. Di bidang *cheating detection*, *superpixel* juga mampu meningkatkan akurasi pengenalan aktivitas mencontek [14]. Fitur yang digunakan untuk mengenali *cheating detection* adalah titik persendian karena aktivitas mencontek hanya berfokus pada pergerakan dari persendian manusia. Titik persendian dideteksi menggunakan MODEC dengan segmentasi superpixel. Hasil deteksi persendian tersebut kemudian digunakan untuk klasifikasi aktivitas mencontek menggunakan *Conditional Random Field* (CRF). Hasil pengujian menunjukkan bahwa akurasi pengenalan aktivitas mencontek dengan *superpixel* secara keseluruhan meningkat dibandingkan dengan tanpa *superpixel* [14]. Namun akurasi *superpixel* meningkat hanya pada persendian pada bagian kepala dan sendi lengan saja, tetapi menurun pada bagian bahu dan pergelangan tangan [14].

Melihat *superpixel* memiliki peranan yang penting dalam performa untuk meningkatkan akurasi, maka dilakukan penelitian mengenai deteksi teks secara otomatis pada citra berbasis *superpixel* menggunakan *Maximally Stable Extremal Regions* dan *Stroke Width Transform*.

II. Studi Literatur

Pada bagian ini berisi teori mengenai *Simple Linear Iterative Clustering*, *Maximally Stable Extremal Regions*, *Edge Detection*, *Stroke Width Transform* dan penelitian-penelitian sebelumnya.

A. Simple Linear Iterative Clustering

Superpixel jauh lebih efisien untuk memproses representasi tingkat tinggi dalam banyak aplikasi *vision* dari pada menggunakan *pixel*. Segmentasi citra ke *superpixel* bertujuan mengelompokkan *pixel* ke daerah homogen yang menangkap citra yang sifatnya redundansi. Gambar 1 memperlihatkan segmentasi citra ke daerah *superpixel*. Representasi *superpixel* memungkinkan untuk dapat mendeskripsikan citra ke tingkat yang lebih tinggi dibandingkan dengan representasi *pixel*. *Simple Linear Interactive Clustering* (SLIC) merupakan teknik yang efektif membagi citra ke daerah homogen (*superpixels*).

SLIC sebenarnya didasarkan pada algoritma klasterisasi *k-means* yang bergantung pada dua parameter, yaitu jumlah *superpixel* yang diinginkan (K) dan parameter *compactness* t. Setiap *pixel* dideskripsikan oleh vector $v = [L, a, b, x, y]$ yang mana terdiri dari ruang warna CIELAB dan lokasi *pixel* (x, y)[10].

Gambar 1. Segmentasi gambar ke daerah *superpixel* menggunakan SLIC [10]

SLIC dimulai dari tahap inisialisasi $|K|$ *vector* pusat *cluster* : $C_1, C_2, \ldots, C_{|K|}$. Untuk *superpixel* sekitar yang berukuran sama akan ada pusat *superpixel* pada setiap interval $S = \sqrt{(N/K)}$, dimana N adalah jumlah dari *pixel* citra. Karena batas spasial dari setiap *superpixel* adalah sekitar S^2 (perkiraan luas dari *superpixel*), maka dapat diasumsikan bahwa *pixel* yang berkaitan dengan pusat *cluster* terletak dalam daerah $2S \times 2S$ sekitar pusat *superpixel* pada bidang xy sehingga hal ini menjadi area pencarian untuk *pixel* terdekat untuk setiap *cluster*. Dengan menggunakan *Euclidean distance*, dapat dihitung jarak D [10].

$$d_{lab} = \sqrt{(l_k - l_i)^2 + (a_k - a_i)^2 + (b_k - b_i)^2} \quad (1)$$

$$d_{xy} = \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2} \quad (2)$$

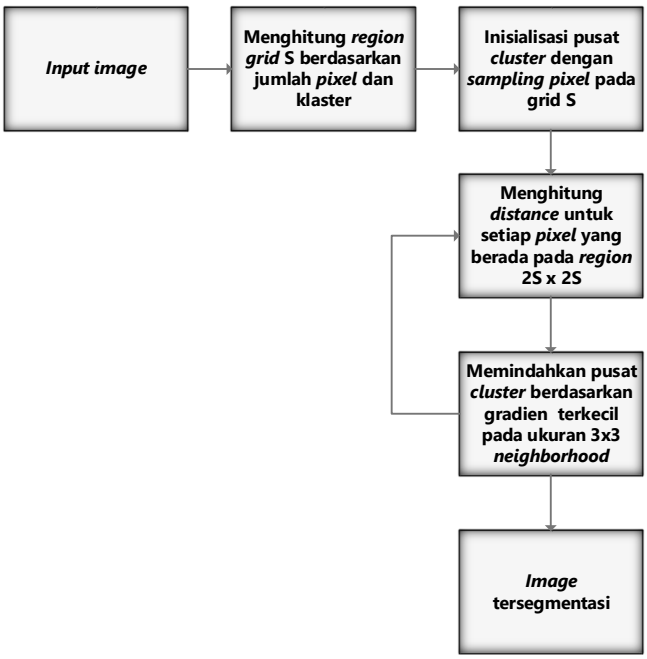
$$D = \sqrt{{d_{lab}}^2 + \left(\frac{d_{xy}}{S}\right)^2 t^2} \quad (3)$$

dimana parameter t berada pada interval [1, 40].

Untuk menghindari penempatan *superpixel* pada lokasi *noise*, pusat *cluster* dipindahkan ke lokasi yang memiliki *gradient* terendah dalam ukuran 3×3 *neighborhood*. *Image gradient* dihitung dengan persamaan (4) berikut.

$$G(x, y) = \|\boldsymbol{I}(x+1, y) - I(x-1, y)\|^2 + \|\boldsymbol{I}(x, y+1) - I(x, y-1)\|^2 \quad (4)$$

Dimana $\boldsymbol{I}(x, y)$ adalah vektor Lab yang berkoresponden terhadap posisi *pixel* (x, y) dan $\|\cdot\|$ adalah normalisasi L_2. Dengan demikian, untuk setiap *pixel* ditentukan pada pusat *cluster* terdekat di wilayah $2S \times 2S$, yang dapat mereduksi jumlah *distance*. Diagram proses SLIC dapat dilihat pada Gambar 2.



Gambar 2. Diagram proses SLIC

Superpixel biasanya digunakan sebagai langkah *preprocessing* dalam metode segmentasi. Metode *superpixel* yang baik adalah metode yang mampu meningkatkan kinerja segmentasinya. Pada penelitian [15] dilakukan perbandingan metode segmentasi yang dihasilkan dari SLIC, GS04, NC05, TP09, QS09, dan GC10 pada dataset MSRC. Hasil ini diperoleh menggunakan *superpixel* untuk menghitung warna, tekstur, geometri, dan fitur lokasi. Kemudian hasilnya dilakukan untuk melatih model klasifikasi untuk 21 kelas objek dengan CRF. Hasilnya menunjukkan bahwa SLIC *superpixel* menghasilkan kinerja terbaik. SLIC juga mengurangi waktu komputasi dengan faktor lebih dari 500 kali lebih baik dari pada NC05. Berdasarkan hasil tersebut, SLIC memiliki tingkat kesalahan segmentasi yang lebih kecil dan kecepatan segmentasi yang lebih cepat dibandingkan metode *superpixel* lainnya.

B. Maximally Stable Extremal Regions

Maximally Stable Extremal Regions (MSER) adalah metode untuk mendeteksi gumpalan (*blob detection*) dalam sebuah citra. MSER didasarkan pada ide dalam mengambil daerah yang hampir sama melalui berbagai nilai ambang (*threshold*) [4]. Semua piksel yang berada di bawah nilai ambang (*threshold*) yang diberikan adalah berwarna putih, selain itu berwarna hitam. Pada urutan citra *threshold* tersebut dengan *frame lt*, dapat dilihat terlebih dulu gambar hitam, kemudian bintik-bintik putih yang sesuai dengan intensitas lokal minima akan muncul, kemudian berkembang lebih besar. Bintik-bintik putih ini pada akhirnya akan bergabung, sampai seluruh gambar putih. Himpunan semua komponen terhubung secara berurutan adalah himpunan semua daerah *extremal* [4].

Sebagai pilihan, dapat dilakukan pembentukan elips dengan proses *fitting*. *Frame* elips yang melekat pada MSER dengan proses *fitting* dilakukan pembentukan elips ke daerah citra yang dituju. Kemudian daerah-daerah tersebut akan disimpan sebagai fitur. Kata *extremal* mengacu pada sifat bahwa semua *pixel* dalam MSER memiliki intensitas baik yang lebih tinggi (daerah *extremal* yang terang) atau lebih rendah (daerah *extremal* yang gelap) dari semua *pixel* pada batas luarnya.

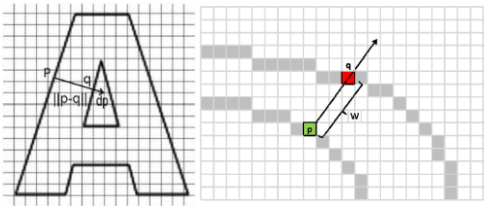
C. Edge Detection

Edge detection atau dikenal sebagai deteksi tepi sering digunakan untuk mendeteksi sebuah objek. Tidak hanya itu, deteksi tepi juga digunakan dalam pengenalan objek pada sebuah citra. Deteksi tepi juga sering digunakan pada tahap *preprocessing* khususnya di bidang *image processing*. Dalam penelitian ini, deteksi tepi digunakan untuk mendeteksi tepi dari *natural image*. Untuk mendeteksi tepi dari sebuah *natural image*, digunakan metode *canny* dan *sobel*. Selanjutnya output dari deteksi tepi akan dikombinasikan dengan MSER untuk meningkatkan hasil wilayah tepi pada *natural image* yang mengandung teks. Tepi yang diperoleh digunakan sebagai batas untuk daerah MSER dan informasi di luar batas tersebut akan dihapus. Pada kombinasi ini, arah gradien dari setiap tepi disesuaikan sehingga dapat memberikan peningkatan representasi teks dalam kondisi *blur*.

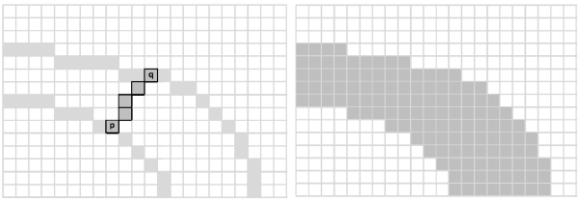
D. Stroke Width Transform

Stroke Width Transform (SWT) merupakan metode untuk menemukan kandidat karakter dalam teks pada sebuah citra khususnya *natural image* [9]. SWT bekerja menggunakan blok-blok yang tersebar di wilayah yang dianggap merupakan kandidat teks pada *natural image*. Proses SWT pada dasarnya dimulai dengan pengelompokkan blok yang akan diproses untuk menentukan wilayah teks. Blok yang mengandung calon karakter (kandidat teks) akan dikelompokkan secara bersama-sama sedangkan blok yang tidak terdeteksi mengandung teks akan dihapus [9]. Metode ini memungkinkan untuk dapat diterapkan pada *natural image* yang mengandung beberapa bahasa serta jenis *font*.

SWT adalah sebuah operator citra lokal yang menghitung lebar *stroke* dimana *stroke* didefinisikan sebagai bagian kontinu dari citra yang membentuk sebuah *band* dengan lebar yang hampir konstan. Lebar yang hampir konstan merupakan salah satu fitur yang membedakan teks dari unsur-unsur lain pada citra, yang dapat dimanfaatkan untuk memulihkan daerah yang mungkin berisi teks. SWT melakukan integrasi informasi secara *bottom-up*, dimana SWT mampu menggabungkan *pixel* lebar *stroke* yang sama ke dalam komponen yang lebih besar. Output dari SWT adalah citra dengan ukuran yang sama dengan ukuran citra input dimana setiap elemen mengandung lebar *stroke* terkait dengan *pixel*. Ilustrasi proses perhitungan lebar *stroke* dengan SWT dapat dilihat pada Gambar 3.



Gambar 3. Proses menentukan lebar *stroke* pada blok dengan SWT [9]



Gambar 4. Pengelompokkan lebar *stroke* dengan *connected component* [9]

Dua *pixel* tetangga akan dikelompokkan bersama-sama jika memiliki lebar *stroke* yang serupa dengan menggunakan *connected component* untuk menemukan kandidat karakter. Proses *connected component* dapat dilihat pada Gambar 4. Nilai awal setiap elemen SWT adalah tak terbatas. Untuk memulihkan *stroke*, dilakukan proses deteksi tepi pada citra menggunakan *canny*. Setelah itu, arah gradien d_p dari setiap tepi *pixel* p dihitung. Jika p terletak pada batas *stroke*, maka d_p harus tegak lurus dengan orientasi *stroke*. Dengan mengikuti persamaan (5), yaitu untuk menentukan blok *ray*,

$$r = p + n.dp \quad (5)$$

untuk $n > 0$ sampai tepi *pixel* lain q ditemukan dan arah gradien di *pixel* q adalah d_q. Jika d_q berlawanan dengan d_p, maka setiap elemen s dari citra output SWT yang sesuai dengan *pixel* segmen $[p, q]$ dilakukan perubahan lebar sebesar $|p - q|$ kecuali telah memiliki nilai yang lebih rendah. Selain itu, jika pencocokan *pixel* q tidak ditemukan, atau jika d_q tidak berlawanan dengan d_p, blok *ray* tersebut akan dibuang [9].

Berikutnya, blok yang tidak terdeteksi ditransformasikan sesuai dengan nilai θ yang terdekat dengan blok terdeteksi secara vertikal/horizontal. Nilai θ yang telah dihitung kemudian ditambahkan dengan nilai θ blok dari arah gradien pada blok yang tidak terdeteksi θ'. Sehingga, sudut baru $(\theta + \theta')$ menjadi arah gradien baru dari kandidat huruf yang tidak terdeteksi setelah proses transformasi [8].

Pada tahap akhir, kandidat huruf dari blok yang terdeteksi dan blok yang tidak terdeteksi dikelompokkan bersama dalam garis teks untuk proses deteksi. Kandidat huruf di blok yang tidak terdeteksi dipilih berdasarkan kesamaan untuk kandidat huruf di blok terdeteksi dengan rasio tinggi huruf dan lebar *stroke* [8]. Kesesuaian tersebut berdasarkan persamaan (6).

$$LC = \begin{cases} Accepted, & SW_{LCU} = SW_{LCD} \; dan \; HR \leq 2.0 \\ Rejected, & otherwise \end{cases} \quad (6)$$

Dimana LC adalah kandidat huruf, SW_{LCU} dan SW_{LCD} masing-masing adalah lebar *stroke* kandidat huruf di blok yang tidak terdeteksi dan blok yang terdeteksi sedangkan HR adalah rasio tinggi huruf.

E. Penelitian Terdahulu

Penelitian [6] mengusulkan *framework* pembelajaran berdasarkan jarak metrik secara terpadu (*unified distance metric learning*) untuk proses klasterisasi hirarkis adaptif (*adaptive hierarchical clustering*). *Framework* ini dirancang untuk mendapatkan bobot kesamaan, yang secara adaptif menggabungkan kesamaan fitur yang berbeda dan *clustering threshold*, yang secara otomatis menentukan jumlah *cluster*. Dengan *framework* tersebut, dibangun sebuah sistem deteksi teks pada sebuah citra (*natural image*) dengan multi orientasi. Proses deteksi teks dimulai dengan membangun wilayah kandidat teks dengan cara mengelompokkan kandidat karakter pada *natural image* berdasarkan klasterisasi adaptif (*adaptive clustering*). Metode yang dilakukan oleh [6] untuk membangun kandidat teks terdiri dari beberapa langkah, yaitu dimulai dengan proses pengelompokkan melalui *single-link clustering* berbasis morfologi, pengelompokkan melalui *divisive hierarchical clustering* berbasis orientasi, dan pengelompokkan melalui *divisive clustering* berdasarkan proyeksi. Pengujian sistem dilakukan pada beberapa *public dataset*, yaitu ICDAR Robust Reading Competition (2011 dan 2013), MSRA-TD500, dan NEOCR. Berdasarkan pengujian sistem, didapatkan *f-measure* sebesar 71%.

Penelitian [2] menggunakan teknik *image partition* untuk mencari kandidat karakter teks berdasarkan fitur lokal gradien dan warna komponen karakter. Kemudian pada penelitian [2] dilakukan pengelompokkan kandidat karakter menjadi sebuah teks yang terdeteksi berdasarkan perbedaan ukuran, jarak, dan perataan karakter.

Penelitian [3] mengemukakan bahwa deteksi teks pada sebuah citra menggunakan teknik *partition text* berdasarkan gradien dan warna kurang efisien. Metode yang diusulkan oleh [3] adalah *ant clustering* untuk digunakan sebagai teknik pengelompokkan dari partisi tersebut. Hal ini menunjukkan bahwa *ant clustering* dapat meningkatkan performa akurasi dari fitur partisi gradien dan warna.

Penelitian [7] mengusulkan fitur simetri untuk deteksi teks pada *natural image*. Dalam sebuah *text line*, kesimetrian dari *intra-character* memiliki korespondensi terhadap kontur dalam dan luar sedangkan simetri *inter-character* membantu dalam mengekstrak wilayah perbedaan antara dua karakter berturut-turut. Pada penelitian [7] digunakan *gradient vector flow* (GVF) untuk mendeteksi kedua jenis simetri ini. Kedua jenis simetri tersebut kemudian dikelompokkan ke dalam *text line* menggunakan konsistensi ukuran, warna, dan *stroke*.

Pada penelitian yang dilakukan oleh [8], digunakan algoritma *Stroke Width Transform* (SWT) untuk mengelompokkan wilayah yang mengandung teks. SWT diterapkan untuk menemukan kandidat karakter dalam teks. Blok yang terdeteksi mengandung kandidat karakter dikelompokkan secara bersama dan blok yang tidak terdeteksi akan dihapus. Sebelum blok yang tidak terdeteksi dihapus, dilakukan proses pemulihan kandidat karakter pada blok tersebut agar wilayah kandidat karakter dapat ditemukan secara optimal. Kandidat karakter huruf dari blok yang terdeteksi dan blok yang tidak terdeteksi dikelompokkan secara bersama untuk mendeteksi sebuah baris teks (*text line*). Kandidat karakter huruf di blok yang tidak terdeteksi dipilih berdasarkan kesamaan untuk kandidat karakter di blok yang terdeteksi sehubungan dengan rasio tinggi huruf dan lebar *stroke*. Hasil deteksi yang didapatkan sebesar 0.60 untuk *precision*, 0.80 untuk *recall*, dan 0.67 untuk *f-measure* untuk dataset ICDAR.

Penelitian [5] menyajikan metode untuk mendeteksi teks dengan mengambil wilayah yang berpotensi mengandung teks sebanyak mungkin menggunakan MSER dan klasterisasi warna untuk mengekstrak komponen yang terhubung (*connected component*). Kemudian, dilanjutkan menggunakan *visual saliency* dan beberapa informasi sebelumnya untuk membedakan wilayah non-teks. Setelah itu dengan proses tersebut, didapatkan wilayah mana yang mengandung teks. Hasil deteksi yang didapatkan sebesar 0.64 untuk *precision*, 0.65 untuk *recall*, dan 0.66 untuk *f-measure* untuk dataset ICDAR.

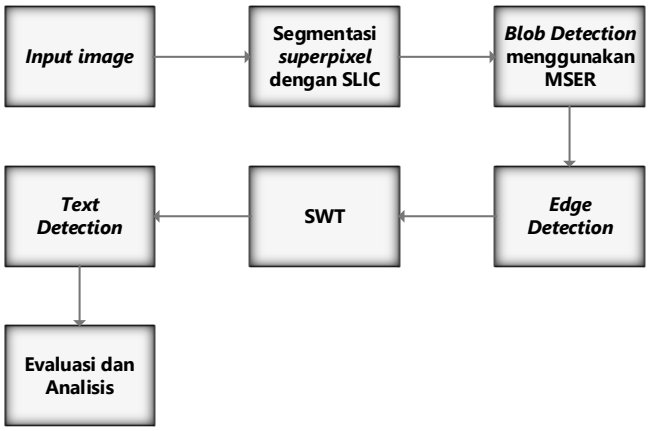
Penelitian [4] mengusulkan algoritma untuk mendeteksi teks dalam *scene image* yang kompleks. Ada dua algoritma yang digunakan, yaitu *Maximally Stable Extremal Regions* (MSER) dan *Stroke Width Transform* (SWT) untuk mendeteksi teks. MSER digunakan untuk mengambil wilayah yang diduga sebagai wilayah yang menonjol kemudian metode *canny* dilakukan untuk peningkatan deteksi tepi pada kandidat teks. *Stroke Width Transform* (SWT) dilakukan juga untuk mengelompokkan wilayah yang merupakan bagian kandidat teks dan yang bukan kandidat teks. Hasil percobaan menunjukkan kinerja yang efektif dari metode tersebut.

Di sisi lain, *superpixel* juga diterapkan di bidang *human action recognition* dalam hal meningkatkan akurasi pengenalan. Penelitian yang dilakukan oleh [12] adalah pengenalan *human action* menggunakan fitur HOG (*histograms of oriented gradients*), HOF (*histograms of optical flow*), dan MBH (*motion boundary histograms*) berbasis *superpixel*. Berdasarkan hasil penelitiannya, pengenalan berbasis *superpixel* lebih baik dibandingkan dengan pengenalan tanpa berbasis *superpixel*. Dengan *superpixel*, akurasi pengenalan terbaik adalah dengan menggunakan fitur HOF sedangkan akurasi pengenalan terendah adalah dengan fitur HOG [12].

Di bidang *human interaction*, *superpixel* juga mampu meningkatkan akurasi. Pada penelitian [13] dikembangkan analisis gerakan berbasis video untuk menentukan apakah dua orang memiliki interaksi dalam sebuah video. Interaksi antara dua orang bisa sangat umum, seperti berjabat tangan, bertukar benda, dan sebagainya. Untuk membuat analisis gerakan yang kuat terhadap *noise*, dilakukan segmentasi pada setiap *frame* video ke satu himpunan *superpixel* dan kemudian memperoleh fitur gerakan dan pola gerakan untuk setiap *superpixel* dengan rata-rata *optical flow* dalam *superpixel* tersebut. Secara khusus, digunakan potongan celah untuk membangun *superpixel* yang konsisten antar *frame* secara spasial maupun temporal. Berdasarkan fitur gerakan dan pola gerakan *superpixel*, [13] mengembangkan sebuah algoritma untuk membagi urutan input video menjadi tiga periode berturut-turut, yaitu dua orang berjalan ke arah satu sama lain, dua orang bertemu satu sama lain, dan dua orang berjalan menjauh dari satu sama lain. Percobaan menunjukkan bahwa algoritma yang diusulkan secara akurat dapat membedakan video dengan dan tanpa interaksi manusia.

III. METODOLOGI

Berdasarkan penelitian mengenai deteksi teks pada *natural image* yang telah dilakukan oleh peneliti sebelumnya, maka pada penelitian ini akan dilakukan deteksi teks pada *natural image* secara otomatis berbasis *superpixel* menggunakan MSER dan SWT. Tahapan yang dilakukan untuk deteksi teks pada sebuah *natural image* dapat dilihat pada Gambar 5.



Gambar 5. Diagram Proses Deteksi Teks pada *Natural Image* Berbasis *Superpixel*

Tahap yang dilakukan pada penelitian ini, yaitu input *natural image*, segmentasi *natural image* dengan *superpixel*, *blob detection* menggunakan MSER di wilayah hasil segmentasi, *edge detection* (deteksi tepi) pada hasil wilayah yang dianggap sebagai kandidat teks, dan validasi wilayah yang merupakan kandidat teks dan non-teks pada *natural image* menggunakan SWT, serta melakukan evaluasi dan analisis dari proses deteksi teks menggunakan SWT.

Metode *superpixel* yang digunakan untuk melakukan segmentasi *natural image* adalah *Simple Linear Iterative Clustering* (SLIC). Kemudian deteksi tepi akan

menggunakan dua metode, yaitu *canny* dan *sobel*. Dua metode tersebut akan dianalisis dan dibandingkan hasilnya. Dataset yang digunakan adalah ICDAR 2003. yang terdiri dari 529 citra. Pengujian dilakukan pada 529 citra tersebut. Hasil pengujian yang dilakukan berupa eksperimen pada segmentasi dengan jumlah *superpixel* berkelipatan 100 [14], dimulai dari 100 sampai dengan 1.000 dan eksperimen deteksi teks menggunakan *superpixel* SLIC dan MSER.

Selanjutnya, hasil wilayah yang dibangun oleh MSER dan SLIC dikombinasikan dengan deteksi tepi untuk meningkatkan hasil tepi kandidat teks. Deteksi tepi menggunakan dua metode, yaitu *canny* dan *sobel*. Dua metode tersebut akan dianalisis dan dibandingkan hasilnya.

IV. Hasil dan Pembahasan

Pada eksperimen ini dilakukan segmentasi pada *natural image* menggunakan *superpixel*. Eksperimen segmentasi dilakukan untuk jumlah *superpixel* tertentu dengan kelipatan 100 sampai dengan 1.000 [14].


Gambar 6a. Segmentasi 100sp


Gambar 6b. Visualisasi 100sp


Gambar 7a. Segmentasi 200sp


Gambar 7b. Visualisasi 200sp


Gambar 8a. Segmentasi 300sp


Gambar 8b. Visualisasi 300sp


Gambar 9a. Segmentasi 400sp

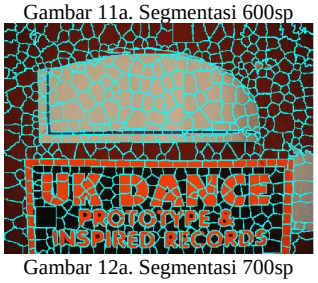

Gambar 9b. Visualisasi 400sp


Gambar 10a. Segmentasi 500sp


Gambar 10b. Visualisasi 500sp


Gambar 11a. Segmentasi 600sp


Gambar 11b. Visualisasi 600sp


Gambar 12a. Segmentasi 700sp

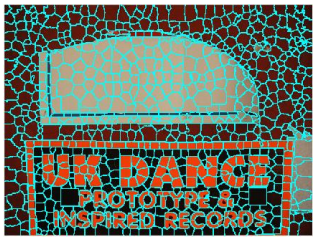

Gambar 12b. Visual 700sp


Gambar 13a. Segmentasi 800sp


Gambar 13b. Visualisasi 800sp


Gambar 14a. Segmentasi 900sp

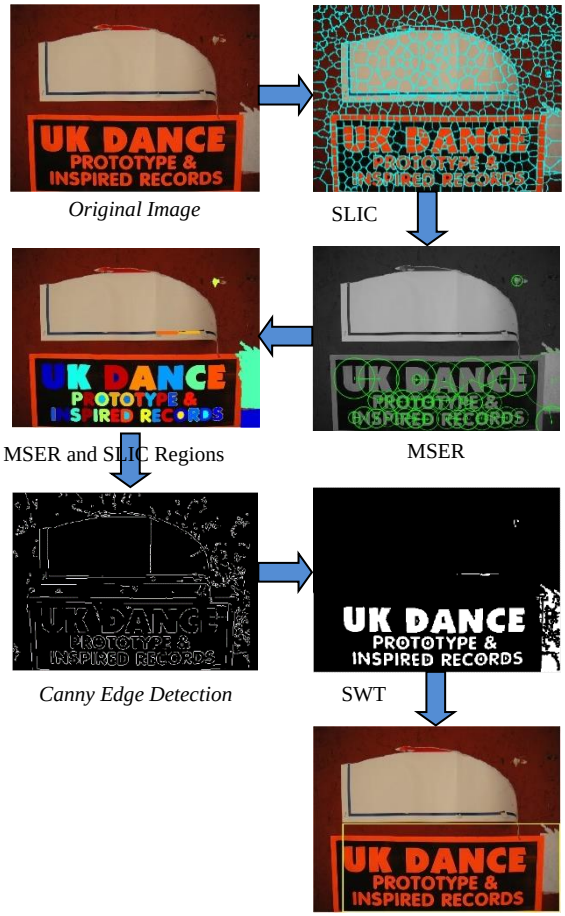

Gambar 14b. Visualisasi 900sp


Gambar 15a. Segmentasi 1.000sp


Gambar 15b. Visualisasi 1.000sp

Gambar 6a, 7a, 8a, 9a, 10a, 11a, 12a, 13a, 14a, dan 15a menunjukkan hasil segmentasi yang dilakukan dengan beberapa jumlah *superpixel* kelipatan 100 sampai dengan 1.000. Sedangkan Gambar 6b, 7b, 8b, 9b, 10b, 11b, 12b, 13b, 14b, dan 15b menunjukkan visualisasi wilayah segmentasi dengan jumlah *superpixel* kelipatan 100 sampai dengan 1.000. Berdasarkan hasil eksperimen terlihat bahwa jumlah *superpixel* sangat berpengaruh terhadap hasil segmentasi pada *natural image*. Variasi segmentasi terlihat pada besarnya jumlah *superpixel*. Semakin besar jumlah *superpixel* yang digunakan maka semakin banyak hasil segmentasi pada wilayah *natural image*.

Berdasarkan eksperimen yang dilakukan, kombinasi MSER dan SLIC yang menggunakan 800 *superpixel* mampu mengambil wilayah yang berpotensi mengandung teks. Proses *edge detection* digunakan untuk melakukan deteksi tepi dari sebuah teks. Kemudian SWT memutuskan mana wilayah teks dan non-teks dari *natural image*. Proses deteksi wilayah teks dapat dilihat pada Gambar 16.



Gambar 16. Proses Deteksi Teks pada *Natural Image*

Deteksi tepi yang diuji ada dua metode, yaitu *canny* dan *sobel*. Saat pengujian dilakukan, deteksi tepi menggunakan *canny* lebih baik dibandingkan dengan *sobel*. Deteksi tepi menggunakan *canny* dan *sobel* dapat dilihat pada Gambar 17 dan 18. Meskipun deteksi tepi menggunakan sobel tidak begitu baik namun metode *Stroke Width Transform* (SWT) mampu memulihkan kembali tepi teks yang hilang sehingga wilayah teks masih dapat dideteksi secara optimal. Proses pemulihan SWT dapat dilihat pada Gambar 19.

Gambar 17. Proses Deteksi Teks pada *Natural Image*

Gambar 18. Deteksi tepi menggunakan sobel dan interseksi MSER dan SLIC terhadap sobel

Gambar 19. Hasil Stroke Width Transform dari canny dan sobel

Hasil deteksi teks pada beberapa *natural image* dapat dilihat pada Gambar 20- 43. Pada Gambar 20, 21, 22, 23, 25, 30, 40, dan 41 menunjukkan bahwa proses deteksi teks berhasil menemukan wilayah teks pada *natural image*. Wilayah teks yang berhasil dideteksi ditunjukkan dengan adanya garis kotak berwarna kuning yang diprediksi sebagai wilayah yang menandakan adanya unsur teks pada bagian dari *natural image*. Beberapa hasil deteksi seperti pada Gambar 27, 31, 33, 34, 35, 36, 42, dan 43 juga tidak berhasil

dideteksi secara sempurna dikarenakan masih banyak aspek yang dinilai erat terhadap unsur teks pada *natural image*.

Hasil deteksi pada *natural image* lain yang masih memiliki kesalahan dalam memprediksi wilayah yang mengandung teks dapat dilihat pada Gambar 24, 26, 27, dan 37. Pada Gambar 24, 26, 27, dan 37 terlihat bahwa unsur teks sudah berhasil diprediksi namun masih terdapat unsur-unsur non teks yang dianggap sebagai kandidat teks sehingga beberapa wilayah yang seharusnya bukan kandidat teks ikut dikelompokkan secara bersama-sama dengan wilayah kandidat teks sehingga terjadi kesalahan prediksi wilayah teks pada *natural image*.

Pada Gambar 28, 29, dan 39 menunjukkan hasil deteksi teks yang masih salah dan tidak sesuai dengan wilayah kandidat teks yang seharusnya, namun masih terdapat wilayah kandidat teks yang dianggap benar. Pada Gambar 31, 32, 33, 34, 35, dan 36 terlihat bahwa proses deteksi teks tidak tepat dikarenakan masih banyak pengaruh warna yang kompleks di sekitar unsur teks yang membuat kesulitan dalam menentukan wilayah kandidat teks pada *natural image*.

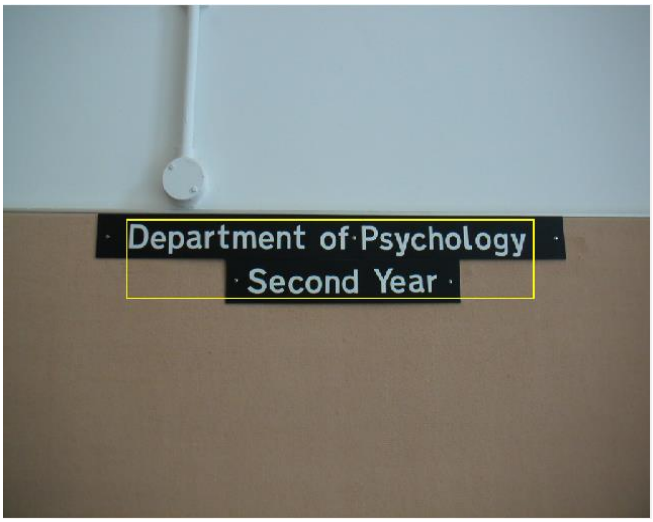
TABEL I
EVALUASI DETEKSI TEKS PADA *NATURAL IMAGE*

Precision	***Recall***	***F-measure***
0,75	0,62	0,68

Evaluasi pengujian yang dilakukan meliputi *precision*, *recall*, dan *f-measure* dapat dilihat pada Tabel I. Secara keseluruhan pengujian, didapatkan *precision*, *recall*, dan *f-measure* masing-masing adalah 0,75; 0,62; dan 0,68. Hasil tersebut secara keseluruhan lebih baik dari pada tanpa *superpixel* namun masih belum baik pada beberapa *natural image*.

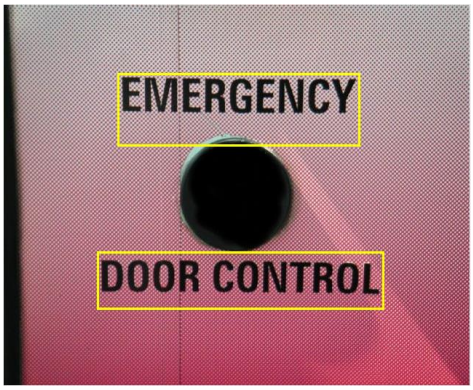
Pada Gambar 42 dan 43 menunjukkan bahwa pencahayaan dan *noise*, serta warna latar (*background*) yang mirip dan hampir menyatu dengan warna teks membuat kesulitan saat proses deteksi teks. Hal ini menjadi faktor penyebab kesalahan dalam proses deteksi pada jenis *natural image* yang kompleks. Beberapa faktor utama dalam mendeteksi teks pada *natural image* adalah warna *background* dan orientasi yang kompleks pada wilayah kandidat teks sehingga proses deteksi masih sulit untuk dilakukan dengan baik.

Gambar 20. Hasil Deteksi Teks pada *natural image*



Gambar 21. Hasil Deteksi Teks pada *natural imge*

Gambar 22. Hasil Deteksi Teks pada *natural image*



e-ISSN : 2443-2229

Gambar 23. Hasil Deteksi Teks pada *natural image*

Gambar 24. Hasil Deteksi Teks pada *natural image*

Gambar 25. Hasil Deteksi Teks pada *natural image*

Gambar 26. Hasil Deteksi Teks pada *natural image*

Gambar 27. Hasil Deteksi Teks pada *natural image*

Gambar 28. Hasil Deteksi Teks pada *natural image*

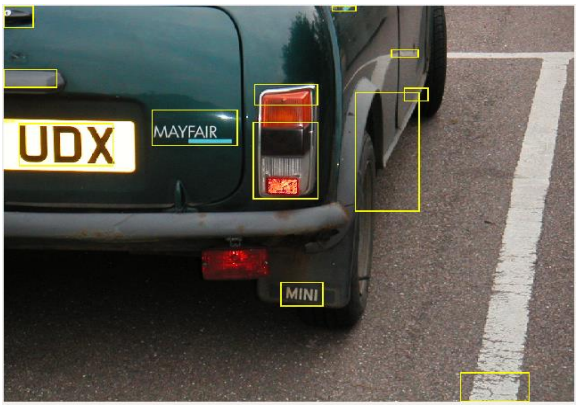
Gambar 29. Hasil Deteksi Teks pada *natural image*

Gambar 30. Hasil Deteksi Teks pada *natural image*

Gambar 31. Hasil Deteksi Teks pada *natural image*

Gambar 32. Hasil Deteksi Teks pada *natural image*

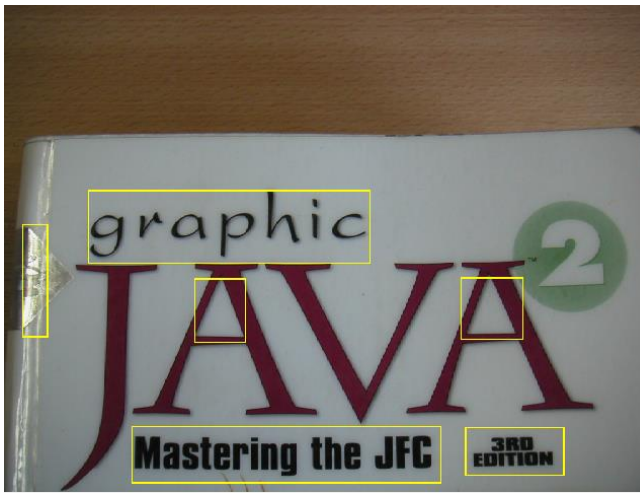
Gambar 33. Hasil Deteksi Teks pada *natural image*



e-ISSN : 2443-2229

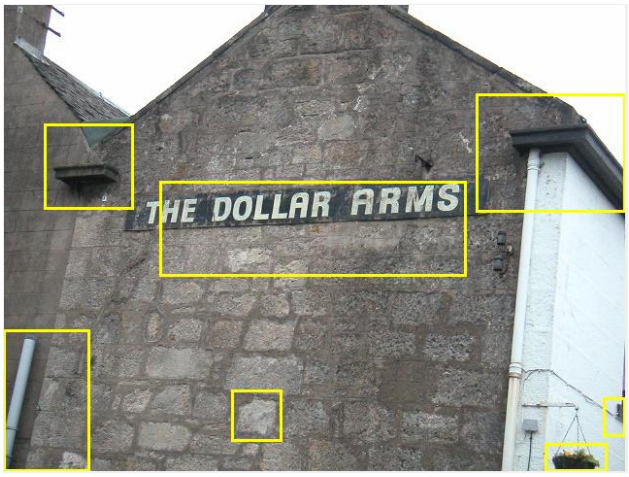
Gambar 34. Hasil Deteksi Teks pada *natural image*

Gambar 35. Hasil Deteksi Teks pada *natural image*



Gambar 36. Hasil Deteksi Teks pada *natural image*

Gambar 37. Hasil Deteksi Teks pada *natural image*



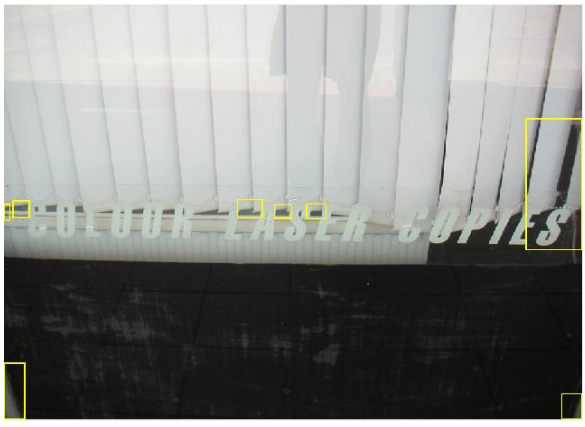
Gambar 38. Hasil Deteksi Teks pada *natural image*

Gambar 39. Hasil Deteksi Teks pada *natural image*

Gambar 40. Hasil Deteksi Teks pada *natural image*

Gambar 41. Hasil Deteksi Teks pada *natural image*

Gambar 42. Hasil Deteksi Teks pada *natural image*



Gambar 43. Hasil Deteksi Teks pada *natural image*

V. KESIMPULAN

Hasil pengujian menunjukkan bahwa MSER dengan *superpixel* mampu mendeteksi teks pada *natural image* dengan SWT. Deteksi wilayah teks pada *natural image* menggunakan *superpixel* lebih baik dibandingkan dengan tanpa *superpixel*. Deteksi tepi menggunakan *canny* lebih baik dibandingkan dengan *sobel*. Namun *Stroke Width Transform* (SWT) mampu memulihkan kembali tepi teks yang hilang pada deteksi tepi *canny* maupun *sobel* sehingga wilayah teks masih dapat dideteksi secara optimal. Hasil pengujian juga menunjukkan bahwa hasil kombinasi MSER dan SLIC dapat mendeteksi teks pada natural image secara lebih baik dibandingkan dengan MSER saja meskipun waktu komputasi lebih lama dibandingkan dengan MSER saja.

DAFTAR PUSTAKA

[1] S. Kumar, K. Kumar, and R. K. Mishra, “Scene Text Recognition using Artificial Neural Network : A Survey,” *Int. J. Comput. Appl. (0975 – 8887)*, vol. 137, no. 6, pp. 40–50, 2016.

[2] C. Yi and Y. Tian, “Text String Detection From Natural Scenes by Structure-Based Partition and Grouping,” *IEEE Trans. Image Process.*, vol. 20, no. 9, pp. 2594–2605, 2011.

[3] P. Tomer and A. Goyal, “Ant clustering based text detection in natural scene images,” *2013 Fourth Int. Conf. Comput. Commun. Netw. Technol.*, pp. 1–7, 2013.

[4] A. Tabassum and S. A. Dhondse, “Text Detection Using MSER and Stroke Width Transform,” *2015 Fifth Int. Conf. Commun. Syst. Netw. Technol.*, pp. 568–571, 2015.

[5] X. Huang, T. Shen, R. Wang, and C. Gao, “Text detection and recognition in natural scene images,” *Int. Conf. Commun. Signal Process. ICCSP 2014 - Proc.*, no. ICEDlF, pp. 1068–1072, 2015.

[6] X.-C. Yin, W.-Y. Pei, J. Zhang, and H.-W. Hao, “Multi-Orientation Scene Text Detection with Adaptive Clustering,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 4, pp. 1930–1937, 2015.

[7] T. Q. Phan, P. Shivakumara, and C. L. Tan, “Detecting text in the real world,” *Proc. 20th ACM Int. Conf. Multimed. - MM ’12*, p. 765, 2012.

[8] J. Jameson and S. N. H. S. Abdullah, “Extraction of Arbitrary Text in Natural Scene Image based on Stroke Width Transform,” pp. 124–128, 2014.

[9] B. Epshtein, E. Ofek, and Y. Wexler, “Detecting text in natural scenes with stroke width transform,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, no. d, pp. 2963–2970, 2010.

[10] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, “SLIC Superpixels,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, Nov. 2010.

[11] L. Xie, H. Zhang, C. Wang, M. Liu, and B. Zhang, “Superpixel-Based PolSAR Images Change Detection,” *IEEE Trans. Geosci. Remote Sens.*, vol. 2, no. 9, pp. 792–796, 2015.

[12] X. Dong, A. Tsoi, and S. Lo, “Superpixel appearance and motion descriptors for action recognition,” in *2014 International Joint Conference on Neural Networks (IJCNN)*, 2014, pp. 1173–1178.

[13] P. Zheng, Y. Cao, and S. Wang, “Motion analysis for human interaction detection using optical flow on lattice superpixels,” *Wuhan Univ. J. Nat. Sci.*, vol. 18, no. 2, pp. 109–116, Apr. 2013.

[14] Yohannes, V. Ayumi, and M. I. Fanany, “Multimodal Decomposable Models by Superpixel Segmentation and Point-in-Time Cheating Detection,” *ICACSIS*, pp. 391–396, 2016.

[15] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, “SLIC superpixels compared to state-of-the-art superpixel methods,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2281, 2012.