Penggunaan Python Sebagai Pengolahan Citra Untuk Mencari Ukuran Partikel Dari Suatu Gambar Mikroskop Elektron

1st Fakhry Abdusysyahid
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
fakhryyy@student.telkomuniversity
acid

2nd Indra W F,
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
indrafathonah@telkomuniversity.ac
.id

3rd Casmika S
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
casmika@telkomuniversity.ac.id

***Abstrak* – Pengolahan citra telah menjadi bidang yang semakin penting dalam berbagai aplikasi, seperti aplikasi untuk tunanetra, tracking objek, keamanan lingkungan, pertanian, kedokteran dan pengolahan data partikel berukuran mikrometer sampai nanometer dari gambar SEM. Perkembangan ini memungkinkan penggunaan metode yang lebih canggih dan efisien untuk mengukur ukuran, bentuk, dan distribusi partikel secara akurat. Pada penelitian ini, akan dilakukan pengembangan sebuah sistem menggunakan pemrograman Python untuk mengolah citra digital dengan tujuan menentukan ukuran partikel dalam sampel. Tujuan utamanya adalah untuk memperoleh informasi yang akurat dan dapat diandalkan tentang distribusi ukuran partikel dengan memanfaatkan kekuatan Python dalam pemrosesan citra dan analisis data.**

***Kata kunci* – Pengolahan Citra, Pengukuran Partikel, *Scanning Electron Microscope* (SEM), *Python*, *OpenCV*.**

I. PENDAHULUAN

Pengolahan citra telah menjadi bidang yang semakin penting dalam berbagai aplikasi, seperti aplikasi pertanian, kedokteran dan pengolahan data partikel berukuran mikrometer dari gambar Scanning Electron Microscope (SEM) [1-6]. Perkembangan dalam pengolahan citra telah mengalami kemajuan pesat sejak awalnya diperkenalkan pada tahun 1960-an. Perkembangan komputer yang semakin canggih memungkinkan para peneliti untuk mengembangkan berbagai algoritma dan metode untuk memproses gambar secara digital dengan lebih efisien.

Perkembangan dalam pengolahan citra telah memainkan peran penting dalam pengukuran partikel [4,5]. Perkembangan ini memungkinkan penggunaan metode yang lebih canggih dan efisien untuk mengukur ukuran, bentuk, dan distribusi partikel secara akurat. Hal ini membuka peluang baru dalam berbagai aplikasi seperti industri farmasi, kimia, dan lingkungan di mana pengukuran partikel menjadi krusial untuk pemahaman dan pengendalian proses.

Pada perkembangan sebelumnya, ada sebuah aplikasi ImageZ yang merupakan aplikasi pengolah citra digital yang memanfaatkan gambar partikel dan untuk mendistribusi diameter partikel tersebut. Kekurangan dari aplikasi ini adalah proses pengukuran tersebut dilakukan secara manual pada tiap-tiap partikel dari gambar. Hal ini kurang efisien sehingga proses pengolahan citra sebagai pengukuran akan membutuhkan waktu yang lama.

Pada penelitian ini, akan dilakukan pengembangan sebuah sistem menggunakan pemrograman Python untuk mengolah citra digital untuk menentukan ukuran semua partikel dalam sampel secara otomatis. Python sebagai alat untuk pengolahan citra telah menarik perhatian karena keunggulannya dalam ketersediaan berbagai pustaka dan framework yang kuat seperti OpenCV, scikit-image, dan PIL (Python Imaging Library). Sehingga, penggunaan Python sebagai alat untuk mengolah citra dapat menjadi solusi yang efisien dan efektif dalam mencari statistik dari suatu gambar partikel [7-13]. OpenCV diakui sebagai salah satu metode paling efisien dan memiliki beragam fitur untuk *computer vision*. Sistem akan dirancang untuk mengidentifikasi partikel kecil dalam citra, mengukur dimensi mereka, dan menganalisis distribusi ukuran partikel dalam sampel. Tujuan utamanya adalah untuk memperoleh informasi yang akurat, efektif, efisien, dan dapat diandalkan tentang distribusi ukuran partikel dengan memanfaatkan kekuatan Python dalam pemrosesan citra dan analisis data.

II. KAJIAN TEORI

Adapun teori yang berkaitan dengan penelitian yang digunakan, yaitu sebagai berikut.

A. Pengolahan Citra

Pengolahan citra merupakan salah satu cabang ilmu dalam bidang informatika yang bertujuan untuk memperbaiki kualitas citra agar lebih mudah dipahami oleh manusia maupun komputer [1-3]. Pengolahan citra adalah sistem di mana proses terjadi dengan menggunakan gambar sebagai masukan dan menghasilkan gambar sebagai keluaran. Ini melibatkan manipulasi foto atau gambar bergerak untuk meningkatkan kualitasnya. Setiap objek dalam gambar memiliki karakteristik unik yang dapat dianalisis secara matematis, seperti warna, tekstur, atau bentuknya. Secara ilmiah, citra dapat dianggap sebagai

representasi tiga dimensi dari suatu fungsi, dengan intensitas warna diekspresikan dalam domain spasial x dan y. Pada tingkat komputasi, warna sering diwakili sebagai angka dalam format RGB, memungkinkan proses digitalisasi citra. Informasi digital ini dapat dimanfaatkan untuk mengelompokkan atau mengelompokkan objek sesuai dengan ciri-ciri yang dimilikinya.

B. Scanning Electron Microscope (SEM)

Scanning Electron Microscope (SEM) adalah teknologi mikroskop elektron yang mampu menghasilkan gambar permukaan sampel dengan resolusi tinggi, memungkinkan pengamatan morfologi sampel pada perbesaran yang tinggi. Prinsip kerjanya melibatkan penggunaan berkas elektron yang dipantulkan dari
permukaan objek yang diamati, dengan komponen utama seperti electron gun untuk menghasilkan berkas elektron, lensa elektromagnetik untuk mengarahkan berkas, dan detektor untuk merekam sinyal elektron. Sinyal elektron yang dihasilkan, seperti secondary electron (SE) dan backscattered electron (BSE), digunakan untuk membentuk citra SEM. Teknologi SEM memungkinkan pemindaian area luas dan pengumpulan data besar untuk analisis karakteristik sampel, termasuk distribusi ukuran.

C. Python

Python merupakan bahasa program interpretatif serbaguna yang menekankan pada penggabungan kapabilitas, kemampuan dengan sintaksis kode yang jelas, serta didukung dengan pustaka standar yang luas [7-10]. Python dikembangkan secara khusus untuk memudahkan pembacaan kode, Python dirancang sebagai alat umum yang memungkinkan pembuatan aplikasi canggih dengan kode yang sederhana. Keunggulan Python tercermin pada penggunaannya dalam berbagai bidang AI seperti pemrograman bahasa alami, data mining, machine learning, dan pengolahan citra, yang didukung oleh kelengkapan pustaka yang dimilikinya.

D. OpenCV

OpenCV, singkatan dari Open Source Computer Vision Library, adalah sebuah perangkat lunak yang dirancang untuk mengolah citra secara real time [11-13]. Awalnya dikembangkan oleh Intel, sekarang pustaka ini didukung oleh Willow Garage dan Itseez. Tujuannya adalah memberikan kemampuan kepada komputer untuk memproses visual seperti manusia. OpenCV tersedia di bawah lisensi BSD yang memberikan kebebasan penuh untuk penggunaan komersial tanpa perlu mengungkapkan kode sumbernya. Kompatibel dengan berbagai bahasa pemrograman seperti C, C++, Java, Python, dan mendukung platform seperti Windows, Linux, Mac OS, iOS, dan Android. OpenCV diakui sebagai salah satu metode paling efisien dan memiliki beragam fitur untuk computer vision. Komunitas OpenCV yang besar dan aktif menyediakan sumber daya, tutorial, dan dukungan yang melimpah.

Pada Python dengan pustaka OpenCV, pengolahan citra untuk pengukuran gambar partikel umumnya terdiri dari beberapa tahapan, yang pertama adalah pembacaan atau pemanggilan sumber gambar oleh program dari komputer dengan kode berikut,

```
image = cv2.imread(image_path)                    (1)
```

cv2.imread merupakan fungsi OpenCV untuk membaca *file* sumber gambar dari komputer oleh program dan (image_path) adalah alamat *file* sumber gambar pada komputer. Setelah dibaca, umumnya gambar kemudian diubah menjadi berskala abu-abu agar gambar dapat membedakan objek gambar dengan latar dalam gambar yang dinamakan gambar biner. Namun, dalam penelitian ini menggunakan sumber gambar dari SEM yang sudah berskala abu-abu. Lalu, gambar dikurangi *noise*-nya menggunakan fungsi Gaussian dengan kode berikut,

```
blurred = cv2.GaussianBlur(image, (5, 5), 0)      (2)
```

cv2.GaussianBlur adalah fungsi OpenCV untuk memberi keburaman agar gambar menjadi halus dan mengurangi *noise*-nya. Kernel Gaussian dengan nilai (5, 5) adalah matriks 5x5 yang menentukan area piksel yang diburamkan, dengan pusat buram di tengah dan efek buram yang berkurang menjauh dari pusat. Kernel ini berukuran tuple, yang berarti memiliki lebar dan tinggi seperti matriks dalam satuan piksel dengan nilai harus sama dan ganjil, sehingga misalkan ukuran 5x5 berarti ada 25 piksel. Semakin besar ukuran kernel, semakin kuat buram yang dihasilkan.

Nilai 1 adalah sigmaX (bernilai 1) dan sigmaY (bernilai 0) adalah deviasi standar Gaussian di arah sumbu-X dan sumbu-Y. Jika sigmaY diset ke 0, maka sigmaY akan sama dengan sigmaX, membuat distribusi Gaussian simetris dengan deviasi standar yang sama di kedua arah, menghasilkan blur yang merata. Sebaliknya, jika sigmaX dan sigmaY berbeda, distribusi Gaussian menjadi elips, menyebabkan blur yang lebih lebar dalam satu arah dibandingkan arah lainnya. Setelah itu, gambar akan dideteksi tepinya menggunakan fungsi Canny dengan kode berikut,

```
edges = cv2.Canny(blurred, 50, 150)               (3)
```

cv2.Canny adalah fungsi OpenCV untuk mendeteksi berbagai tepi dalam gambar. Nilai 50 dan 150 berfungsi sebagai ambang batas bawah (threshold1) dan atas (threshold2) dalam deteksi tepi gambar partikel. Pada gambar terdapat dua tepi gambar, yaitu tepi lemah dan tepi kuat. Tepi kuat adalah tepi yang jelas dan tegas, dan biasanya merupakan bagian dari objek yang jelas atau kontur penting dalam gambar. Tepi lemah adalah tepi kurang jelas dan bisa menjadi bagian dari detail yang kurang signifikan atau noise dalam gambar. Threshold1 menandai tepi lemah yang nilai gradiennya berada di antara threshold1 dan threshold2, tetapi hanya dianggap valid jika terhubung dengan tepi kuat. Threshold2 menandai tepi kuat dengan nilai gradien di atas nilai ini, yang selalu dipertahankan karena dianggap signifikan. Dengan cara ini, sistem menyaring noise dan tepi yang kurang penting, mempertahankan hanya tepi yang jelas dan relevan.

Jika nilai threshold2 tinggi akan mengurangi jumlah tepi yang terdeteksi namun meningkatkan akurasi, sedangkan jika nilai threshold1 rendah akan mendeteksi lebih banyak tepi, termasuk tepi lemah, tetapi berisiko

mencakup noise. Nilai threshold1 tinggi mengurangi jumlah tepi lemah yang dipertahankan, sehingga lebih banyak tepi signifikan yang diidentifikasi dan mengurangi noise. Nilai gradien piksel adalah perubahan intensitas warna atau kecerahan pada gambar. Gradien ini memberikan informasi tentang seberapa cepat dan dalam arah mana intensitas gambar berubah dari satu piksel ke piksel lainnya. Kemudian partikel dalam gambar dicari konturnya menggunakan kode berikut,

```
contours = cv2.findContours (edges,
  cv2.RETR_EXTERNAL,
  cv2.CHAIN_APPROX_SIMPLE)        (4)
```

cv2.findContours adalah fungsi OpenCV untuk mencari kontur tiap partikel dalam gambar. cv2.RETR_EXTERNAL berfungsi untuk memastikan bahwa hanya kontur terluar yang diambil. cv2.CHAIN_APPROX_SIMPLE berfungsi untuk memastikan bahwa titik kontur disimpan secara efisien. Terakhir, tiap partikel dalam gambar akan dicari diameternya dengan kode berikut,

```
(x, y), radius = cv2.minEnclosingCircle(contour)
        diameter_pixels = radius * 2        (5)
```

cv2.minEnclosingCircle adalah fungsi OpenCV untuk menghitung lingkaran terkecil yang dapat melingkupi semua titik kontur. Lalu, karena gambar merupakan gambar digital, diameter partikel masih bersatuan piksel. Oleh karena itu, diameter diubah dari piksel menjadi mikrometer dengan kode berikut,

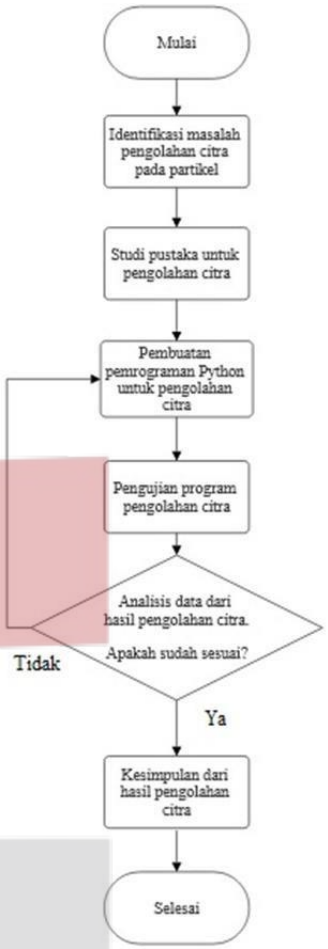
```
diameter_micrometers =
(diameter_pixels * scale) / deltax        (6)
```

scale merupakan skala gambar pada sumber gambar. deltax adalah selisih dari piksel dalam sumbu-x dari garis skala gambar pada sumber gambar (Δx piksel).

III. METODE

A. Diagram Alir Penelitian

Pada tahap awal, penelitian dimulai dari identifikasi masalah dalam pengolahan citra dari partikel. Studi pustaka OpenCV dilakukan untuk dipelajari bagaimana OpenCV melakukan pengolahan citra dalam pemrograman Python. Kemudian, dilanjutkan dengan pembuatan pemrograman Python untuk mengolah citra dari gambar partikel SEM yang disediakan. Pada bagian ini terjadi beberapa fungsi dalam pemrograman. Fungsi pertama adalah fungsi untuk membaca gambar partikel SEM yang akan digunakan dari komputer. Selanjutnya adalah fungsi untuk mengolah gambar tersebut sehingga ditemukan hasil pengukuran dari partikel. Lalu, hasil dari pengolahan citra tersebut ditampilkan dalam sebuah plot grafik berisi hasil data diameter partikel dalam gambar tersebut. Setelah itu, maka akan dilakukan analisis dan validasi hasil data pengukuran dengan membandingkan dengan aplikasi ImageZ. Jika hasil pengukuran tidak sama, maka dilakukan lagi pengolahan citra dengan mengubah variasi nilai dan fungsi-fungsi dalam program. Terakhir, memberi kesimpulan dari penggunaan pemrograman pengolah citra Python. Proses diatas digambarkan pada Gambar 1.

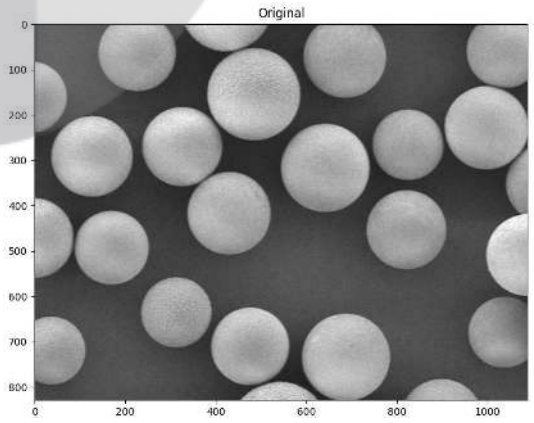


GAMBAR 1.
Diagram Alir Pengerjaan Tugas Akhir

B. Simulasi Pengolahan Citra

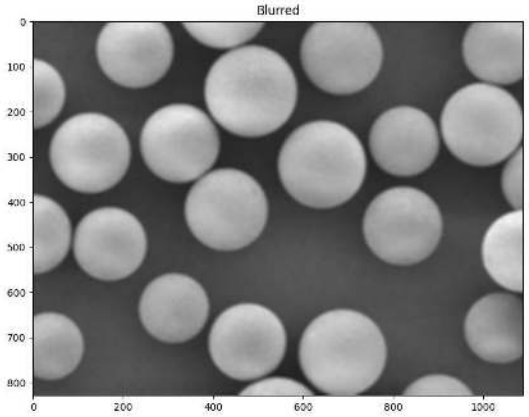
Berikut merupakan proses simulasi pemrograman pengolahan citra dengan menggunakan Python :

1. Pada pemrograman ini, pengolahan citra dimulai dari pembacaan sumber gambar partikel dari komputer. Setelah dibaca, gambar ditampilkan dalam sebuah plot untuk mencari skala (Gambar 2).



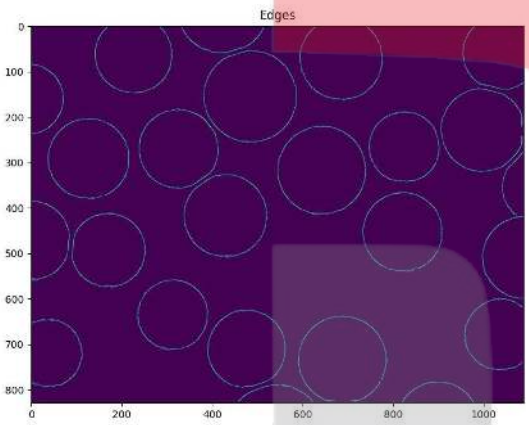
GAMBAR 1.
Sumber Gambar

2. Lalu, gambar akan diburamkan untuk dikurangi *noise*-nya (Gambar 3),



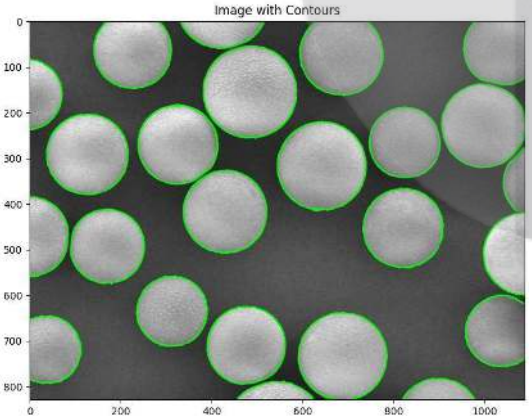
GAMBAR 2.
Gambar diburamkan

3. lalu setiap partikel akan dicari tepiannya (Gambar 4),



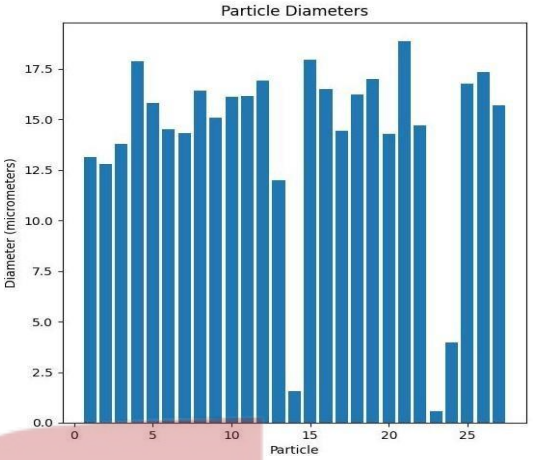
GAMBAR 3.
Tepian Partikel dalam Gambar

4. dan akan diberi kontur untuk memisahkan masing-masing partikel dari partikel lainnya (Gambar 5).



GAMBAR 4.
Kontur Pada Partikel dalam Gambar

5. Setelah itu, setiap partikel akan dilakukan pengukuran untuk dicari diameter terkecil, diameter terbesar, dan juga rata-rata diameter partikel. Pengukuran tersebut dilakukan dengan membandingkan skala yang tertera pada gambar dengan piksel dari gambar. Hasil pengukuran tersebut kemudian ditampilkan dalam sebuah plot grafik (Gambar 6).



GAMBAR 5.
Plot Grafik Hasil Pengolahan Citra

C. Pengukuran dan Validasi Hasil

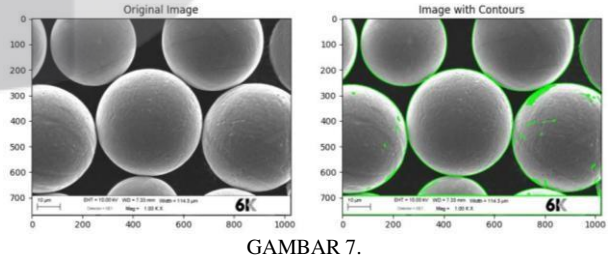
Setelah ditemukan hasil dari pengolahan citra dari gambar partikel, data hasil diameter partikel akan dilakukan pembuangan outlier untuk membuang data diameter selain partikel. Langkah ini dilakukan agar data hasil lebih akurat. Kemudian, data hasil akan dilakukan pembandingan dengan aplikasi ImageZ dari sumber gambar untuk memvalidasi hasil. Hal ini dilakukan untuk melihat seberapa efektif dan efisien pemrograman pengolahan citra Python dibandingkan dengan aplikasi ImageZ.

IV. HASIL DAN PEMBAHASAN

A. Hasil dari Pengkonturan Gambar

Pada proses awal penelitian dilakukan pemilihan citra atau gambar SEM. Gambar uji coba yang akan dipilih dilihat dari banyaknya jumlah partikel dalam gambar. Hal ini dilakukan agar pengaturan nilai fungsi pengkontur gambar dalam program bisa distandarisasi, sehingga program bisa mengkontur tiap partikel berapapun jumlahnya. Pada proses ini juga dilakukan pencarian skala gambar dengan cara mencari Δx piksel dari garis skala gambar dalam sumber gambar.

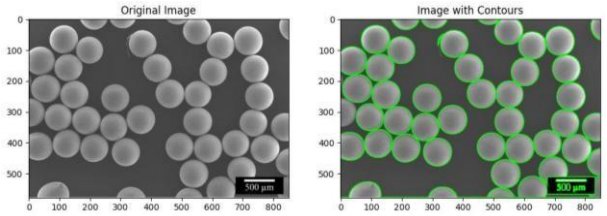
1. Gambar Partikel Pertama



GAMBAR 7.
Gambar SEM Partikel Bubuk

Gambar 7 merupakan gambar uji coba pertama yaitu gambar SEM dari sebuah partikel bubuk. Gambar ini digunakan untuk uji coba pertama karena memiliki jumlah partikel yang sedikit dan memiliki gambar partikel yang besar. Hal ini dilakukan untuk melihat bahwa gambar bisa di beri kontur yang bagus di tiap partikelnya. Pada gambar ini didapat Δx piksel bernilai 90 piksel.

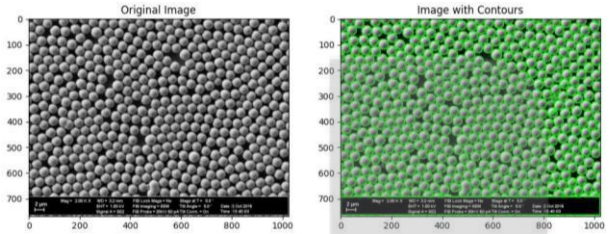
2. Gambar Partikel Kedua



Gambar 8. Gambar SEM Partikel Gelas

Gambar 8 merupakan gambar SEM dari sebuah partikel gelas. Gambar ini memiliki jumlah partikel yang lebih banyak dan ukuran partikel yang lebih kecil dari gambar pertama. Hal ini menunjukan bahwa program bisa memberi kontur pada tiap partikel dengan bagus. Pada gambar ini didapat Δx piksel bernilai 103 piksel.

3. Gambar Partikel Ketiga



Gambar 9. Gambar SEM Partikel PMMA

Gambar 9 merupakan gambar SEM dari sebuah partikel PMMA atau yang lebih dikenal sebagai akrilik. Gambar ini memiliki jumlah partikel yang lebih banyak, ukuran lebih kecil, dan lebih rapat dari gambar sebelumnya. Hal ini menunjukan bahwa program bisa memberi kontur pada tiap partikel dengan bagus. Pada gambar ini didapat Δx piksel bernilai 36 piksel.

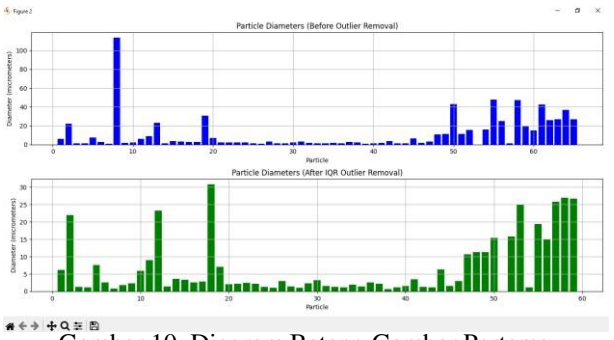
Ketiga percobaan ini menghasilkan nilai standar untuk fungsi OpenCV cv.Canny dengan nilai ambang bawah dan atas bernilai 75 dan 200. Sehingga fungsi dalam program menjadi,

$$edges = cv2.Canny(blurred, 75, 200) \quad (7)$$

B. Hasil Pengukuran Gambar

1. Diagram Batang dan Statistik Partikel

Berikut merupakan diagram batang yang menunjukkan hasil dari pengukuran diameter tiap partikel dan statistik dari ketiga gambar, dengan sumbu-x menunjukkan masing – masing partikel dari partikel-0 ke partikel-n dan sumbu-y menunjukkan diameter dari partikel tersebut.

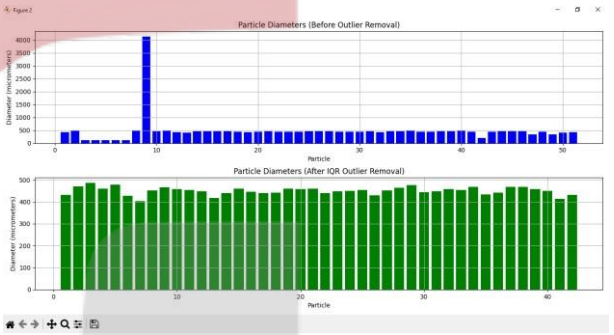


Gambar 10. Diagram Batang Gambar Pertama

```
Basic Particle Statistics (Before Removing Outliers):
Number of particles: 65
Smallest particle diameter: 0.00 micrometers
Largest particle diameter: 113.73 micrometers
Average particle diameter: 11.15 micrometers

Particle Statistics After Removing Outliers (IQR Method):
Number of particles: 59
Smallest particle diameter: 0.00 micrometers
Largest particle diameter: 30.78 micrometers
Average particle diameter: 6.67 micrometers
```

Gambar 11. Statistik Gambar Pertama

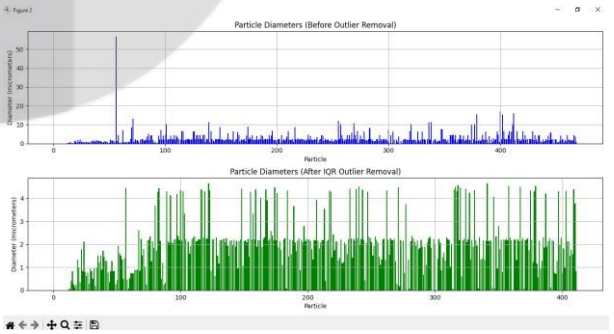


Gambar 12. Diagram Batang Gambar Kedua

```
Basic Particle Statistics (Before Removing Outliers):
Number of particles: 51
Smallest particle diameter: 108.66 micrometers
Largest particle diameter: 4134.17 micrometers
Average particle diameter: 480.02 micrometers

Particle Statistics After Removing Outliers (IQR Method):
Number of particles: 42
Smallest particle diameter: 403.96 micrometers
Largest particle diameter: 485.85 micrometers
Average particle diameter: 450.04 micrometers
```

Gambar 13. Statistik Gambar Kedua



Gambar 14. Diagram Batang Gambar Ketiga

```
Basic Particle Statistics (Before Removing Outliers):
Number of particles: 468
Smallest particle diameter: 0.00 micrometers
Largest particle diameter: 56.74 micrometers
Average particle diameter: 2.82 micrometers

Particle Statistics After Removing Outliers (IQR Method):
Number of particles: 411
Smallest particle diameter: 0.00 micrometers
Largest particle diameter: 4.64 micrometers
Average particle diameter: 1.96 micrometers
```

GAMBAR 15.
Statistik Gambar Ketiga

2. Analisis Diagram dan Tabel Hasil

Berdasarkan diagram dan statistik setiap gambar, program pengolahan citra menggunakan Python dapat menghasilkan pengukuran gambar SEM partikel yang cukup bagus. Sebaran ukuran diameter pada diagram sangat dekat dengan diameter rata–rata partikel. Hal ini menunjukkan bahwa program dapat mengukur diameter partikel dengan cukup akurat.

Adapun kekurangan yang terjadi adalah program tidak dapat mengukur partikel terkecil dan terbesar secara akurat. Hal ini terjadi akibat program memberi kontur pada setiap tepian yang ada dalam gambar SEM, seperti tulisan, angka, simbol, dan lainnya. Sumber gambar SEM yang digunakan juga mempunyai *noise* pada partikel dalam gambar, seperti permukaan partikel tidak halus, bagian partikel yang terpotong, dan partikel yang menyatu. Hal ini membuat program memberi kontur selain partikel, sehingga membuat program sulit mencari diameter partikel terkecil dan terbesar secara akurat.

Solusi yang dapat dilakukan adalah dengan membersihkan hasil data diameter dari data diameter objek selain partikel seperti noise, angka, tulisan, dan lainnya, yang dinamakan data outlier. Outlier adalah data yang berbeda secara signifikan dari data lainnya dan dapat mengganggu analisis, dalam hal ini adalah data dari noise dan lainnya, untuk mendapatkan data ukuran diameter yang lebih akurat dari partikel dalam gambar SEM.

Metode statistika Interquartile Range (IQR) mengukur sebaran data dan mendeteksi pencilan dengan fokus pada 50% data tengah. IQR dihitung sebagai selisih antara kuartil ketiga (Q3) dan kuartil pertama (Q1). Kuartil pertama (Q1) adalah batas bawah 25% data terendah, sementara kuartil ketiga (Q3) adalah batas bawah 75% data. Dengan demikian, IQR memberikan rentang data di sekitar median dan membantu mengidentifikasi data yang dianggap pencilan jika berada di luar rentang ini.

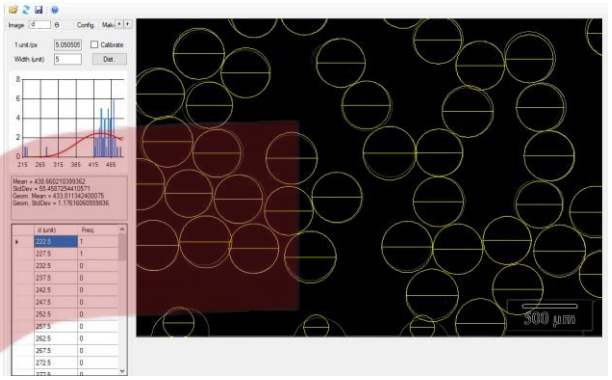
$$Q1 = \text{np.percentile(particle\_diameters, 25)}$$
$$Q3 = \text{np.percentile(particle\_diameters, 75)}$$
$$IQR = Q3 - Q1 \quad (8)$$

$$\text{lower\_bound} = Q1 - 1.5 * IQR$$
$$\text{upper\_bound} = Q3 + 1.5 * IQR \quad (9)$$

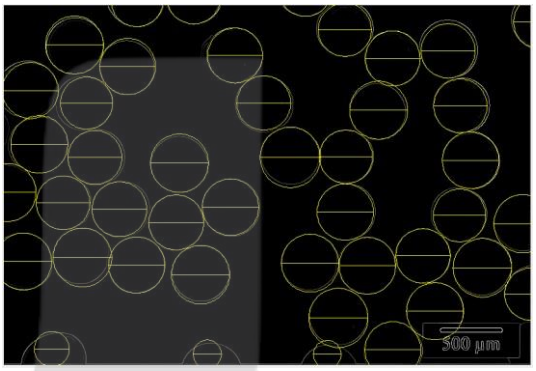
Formula ini digunakan untuk mengidentifikasi nilai yang dianggap sebagai outlier. Nilai yang lebih rendah dari batas bawah atau lebih tinggi dari batas atas dinyatakan sebagai outlier. IQR, yaitu selisih antara Q3 dan Q1, mengukur sebaran data di sekitar median. Nilai 1.5 adalah faktor untuk menentukan seberapa jauh sebuah nilai harus berada dari kuartil untuk dianggap sebagai outlier. Faktor ini secara empiris terbukti efektif dalam banyak kasus untuk mendeteksi data yang tidak biasa tanpa terlalu sensitif.

C. Perbandingan dengan Aplikasi ImageZ

Aplikasi ImageZ adalah aplikasi pengolah citra digital yang memanfaatkan gambar partikel dan fiber untuk mendistribusi diameter partikel dan fiber tersebut. Aplikasi ini digunakan untuk mencari dan mendistribusi ukuran diameter dari partikel dalam gambar SEM. Kekurangan yang terdapat pada aplikasi ImageZ adalah pengguna harus mencari diameter satu persatu partikel secara manual, dengan cara meng-klik tiap partikel yang ingin dicari diameternya.

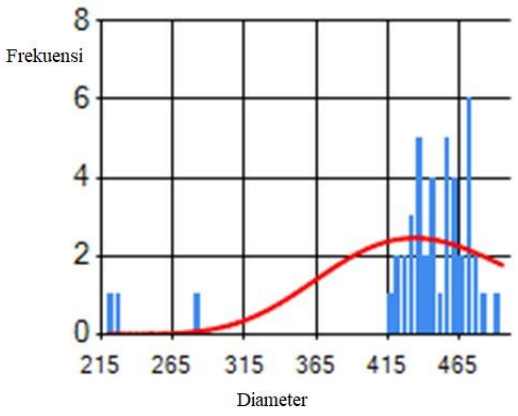


GAMBAR 16.
Aplikasi ImageZ



GAMBAR 17.
Hasil Kontur Gambar Kedua dengan Aplikasi ImageZ

Gambar 16 merupakan penggunaan aplikasi ImageZ, dengan menggunakan sumber gambar partikel SEM kedua sebelumnya (Gambar 8). Penggunaan aplikasi ImageZ dilakukan dengan cara meng-klik satu per satu partikel dalam gambar, dengan menggunakan tepian Canny dan threshold 25 (semakin tinggi threshold, pengkonturan akan semakin tajam sehingga kontur menjadi hilang). Gambar 17 merupakan hasil dari pengkonturan dengan aplikasi ImageZ.



GAMBAR 18.
Grafik Hasil Gambar Kedua dengan Aplikasi ImageZ

Gambar 18 merupakan grafik hasil dari penggunaan aplikasi ImageZ. Terlihat bahwa frekuensi partikel mayoritas berada dalam rentang 415 – 500 mikrometer. Hasil aplikasi ImageZ juga menunjukkan bahwa diameter rata-rata adalah 438.66 mikrometer, sedangkan pada penggunaan program Python sebelumnya diameter rata-rata nya adalah 450.04 mikrometer. Jika dihitung nilai error nya dengan rumus,

(10)

Nilai error dari program Python adalah 2.59%. Hal ini menunjukkan bahwa program pengolahan citra Python cukup akurat dengan aplikasi ImageZ karena nilai error yang masih berada dibawah 5%. Sehingga terbukti bahwa program pengolahan citra Python sedikit lebih unggul dibandingkan dengan aplikasi ImageZ, karena mampu mengolah gambar partikel SEM lebih cepat dan hasil pengukuran data yang cukup akurat dengan aplikasi ImageZ.

V. KESIMPULAN

Pemrograman pengolahan citra Python dapat digunakan untuk mencari distribusi ukuran diameter dari semua partikel dalam sebuah gambar SEM secara langsung. Pemrograman pengolahan citra Python memiliki kekurangan dalam mencari ukuran diameter partikel terkecil dan terbesar dikarenakan adanya noise dan objek lain dalam gambar SEM. Penggunaan metode IQR untuk membuang data outlier, seperti noise dan objek lain selain partikel, dapat meningkatkan keakuratan data hasil pengolahan citra Python. Penggunaan pemrograman pengolah citra Python lebih efisien dan efektif dibanding dengan penggunaan aplikasi ImageZ sebagai pengukuran diameter partikel dalam gambar, karena dapat mengukur setiap partikel secara langsung tanpa harus memilih satu persatu partikel dalam gambar juga memiliki nilai error 2.59% dibanding aplikasi ImageZ.

REFERENSI

[1] Permata Sari, D., Rasyad, S., & Evelina. (2017). Identifikasi Huruf Braille Berbasis Image Processing Secara Real Time.

[2] Mulyawan, H., Samsono, M. Z. H., & Setiawardhana. (2011). Identifikasi dan Tracking Objek Berbasis Image Processing Secara Real Time.

[3] Jumadi, J., Yupianti, & Sartika, D. (2021). Pengolahan Citra Digital untuk Identifikasi Objek menggunakan Metode Hierarchical Agglomerative Clustering.

[4] Alvin Fachrully Septiano, Susilo, Natalia Erna Setyaningsih. (2021). Analisis Citra Hasil Scanning Electron Microscopy Energy Dispersive X-Ray (SEM EDX) Komposit Resin Timbal dengan Metode Contrast to Noise Ratio (CNR).

[5] Damar Rastri Adhika, Atsarina Larasati Anindya, Viny Veronika Tanuwijaya, Heni Rachmawat. (2018). Teknik Pengamatan Sampel Biologi dan Non-konduktif Menggunakan Scanning Electron Microscopy.

[6] Faruqi, M. A. (2021). SISTEM PEMETAAN POSISI OBJEK KENDARAAN MENGGUNAKAN PENGOLAHAN CITRA PADA AREA 360°.

[7] Reza, M., Al Qossam Maududi, I., Rifki, M., Mujaddid, A., Ikhsanudin, F., Adharani, Y., Ambo, S. N., & Rosanti, N. (2022). Artificial Intelligence: Image Processing & Application with Python.

[8] Muhammad, R., & Yulianto, S. (2022). Penerapan Pemrograman Python dalam Menentukan Waktu Overhaul Kondensor Turbin Uap.

[9] Semendawai, J. N., Febiola, I., Pamungkas, B., & Ruliansyah, M. D. (2021). Perancangan Aplikasi Otomatisasi Menggunakan Bahasa Pemrograman Python Pada Aktivitas Monitoring Pemakaian Data Harian Kartu Internet Of Things.

[10] Sambi Ua, A. M. T. I., H, D. L., Marpaung, E. S. K., Ong, J., Savinka, M., Nurhaliza, P., & Ningsih, R. Y. (2023). Penggunaan Bahasa Pemrograman Python Dalam Analisis Faktor Penyebab Kanker Paru-Paru.

[11] Muchtar, H., & Apriadi, R. (2019). Implementasi Pengenalan Wajah Pada Sistem Penguncian Rumah dengan Metode Template Matching Menggunakan Open Source Computer Vision Library (OpenCV).

[12] Zulkhaidi, T. C. A. S., Maria, E., & Yulianto. (2019). Pengenalan Pola Bentuk Wajah dengan OpenCV.

[13] Andrekha, M. Z., & Huda, Y. (2021). Deteksi Warna Manggis Menggunakan Pengolahan Citra dengan OpenCV Python.