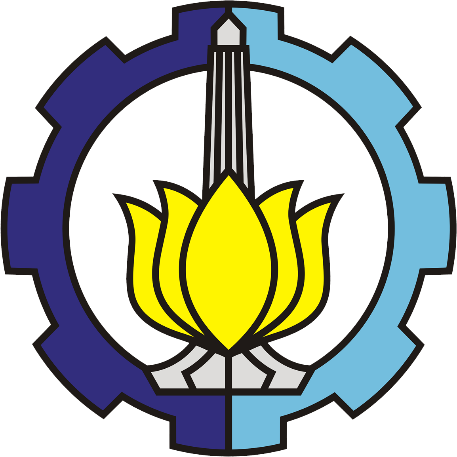
**LAPORAN PENGOLAHAN CITRA MEDIKA**

***Image Labeling* Citra Mikroskopik Sel Darah Merah**

**Mata Kuliah : Pengolahan Citra Medika**

**Dosen : Nada Fitrieyatul Hikmah, S.T., M.T.**



**Disusun oleh :**

**Muhammad Yasin**

**07311740000019**

**DEPARTEMEN TEKNIK BIOMEDIK**

**FAKULTAS TEKNOLOGI ELEKTRO DAN INFORMATIKA CERDAS**

**INSTITUT TEKNOLOGI SEPULUH NOPEMBER**

**2021**

**BAB I**

**TINJAUAN PUSTAKA**

* 1. **Bahasa Pemrograman Python**

Python adalah [bahasa pemrograman](https://id.wikipedia.org/wiki/Bahasa_pemrograman" \o "Bahasa pemrograman) [tujuan umum](https://id.wikipedia.org/w/index.php?title=Bahasa_pemrograman_tujuan_umum&action=edit&redlink=1" \o "Bahasa pemrograman tujuan umum (halaman belum tersedia)) yang [ditafsirkan](https://id.wikipedia.org/w/index.php?title=Bahasa_yang_ditafsirkan&action=edit&redlink=1" \o "Bahasa yang ditafsirkan (halaman belum tersedia)), [tingkat tinggi.](https://id.wikipedia.org/wiki/Bahasa_pemrograman_tingkat_tinggi" \o "Bahasa pemrograman tingkat tinggi) Dibuat oleh [Guido van Rossum](https://id.wikipedia.org/wiki/Guido_van_Rossum) dan pertama kali dirilis pada tahun 1991, filosofi desain Python menekankan [keterbacaan kode](https://id.wikipedia.org/wiki/Pemrograman_komputer" \l "Keterbacaan_kode_sumber" \o "Pemrograman komputer) dengan penggunaan [spasi putih yang signifikan](https://id.wikipedia.org/w/index.php?title=Aturan_off-side&action=edit&redlink=1" \o "Aturan off-side (halaman belum tersedia)). [Konstruksi bahasa](https://id.wikipedia.org/w/index.php?title=Konstruksi_bahasa&action=edit&redlink=1" \o "Konstruksi bahasa (halaman belum tersedia))nya dan pendekatan berorientasi objek bertujuan untuk membantu [programmer](https://id.wikipedia.org/wiki/Programmer) menulis kode yang jelas dan logis untuk proyek skala kecil dan besar. [1]

Python [diketik secara dinamis](https://id.wikipedia.org/wiki/Bahasa_pemrograman_dinamis" \o "Bahasa pemrograman dinamis) dan [pengumpulan sampah](https://id.wikipedia.org/wiki/Pengumpulan_sampah_(ilmu_komputer)" \o "Pengumpulan sampah (ilmu komputer)). Ini mendukung beberapa [paradigma pemrograman](https://id.wikipedia.org/wiki/Paradigma_pemrograman" \o "Paradigma pemrograman), termasuk pemrograman [terstruktur](https://id.wikipedia.org/wiki/Pemrograman_terstruktur" \o "Pemrograman terstruktur) (terutama, prosedural), [berorientasi objek](https://id.wikipedia.org/wiki/Pemrograman_berorientasi_objek" \o "Pemrograman berorientasi objek), dan [fungsional](https://id.wikipedia.org/wiki/Pemrograman_fungsional" \o "Pemrograman fungsional). Python sering dideskripsikan sebagai bahasa "termasuk baterai" karena perpustakaan standarnya yang komprehensif. [2]

Python dikandung pada akhir 1980-an sebagai penerus [bahasa ABC](https://id.wikipedia.org/w/index.php?title=ABC_(bahasa_pemrograman)&action=edit&redlink=1" \o "ABC (bahasa pemrograman) (halaman belum tersedia)). Python 2.0, dirilis pada tahun 2000, memperkenalkan fitur-fitur seperti [pemahaman daftar](https://id.wikipedia.org/w/index.php?title=Pemahaman_daftar&action=edit&redlink=1" \o "Pemahaman daftar (halaman belum tersedia)) dan sistem pengumpulan sampah dengan [penghitungan referensi](https://id.wikipedia.org/w/index.php?title=Penghitungan_referensi&action=edit&redlink=1" \o "Penghitungan referensi (halaman belum tersedia)).

Python 3.0, dirilis pada tahun 2008, adalah revisi utama dari bahasa yang tidak sepenuhnya [kompatibel](https://id.wikipedia.org/w/index.php?title=Kompatibilitas_terbalik&action=edit&redlink=1" \o "Kompatibilitas terbalik (halaman belum tersedia)) dengan versi sebelumnya, dan banyak kode Python 2 yang tidak berjalan tanpa modifikasi pada Python 3.

[Penerjemah](https://id.wikipedia.org/wiki/Penerjemah_(komputasi)) Python tersedia untuk banyak [sistem operasi](https://id.wikipedia.org/wiki/Sistem_operasi" \o "Sistem operasi). Komunitas programmer global mengembangkan dan memelihara CPython, implementasi referensi [3] yang [gratis dan bersumber terbuka](https://id.wikipedia.org/wiki/Free_and_open-source_software). Sebuah organisasi nirlaba, [Python Software Foundation](https://id.wikipedia.org/w/index.php?title=Python_Software_Foundation&action=edit&redlink=1), mengelola dan mengarahkan sumber daya untuk pengembangan Python dan CPython.

* 1. ***Library* yang digunakan pada Pengolahan Citra**
     1. ***Scikit Image***

Scikit-image adalah *library* pemrosesan gambar *open source* untuk bahasa pemrograman Python. [4] Ini mencakup algoritma untuk segmentasi, transformasi geometri, manipulasi ruang warna, analisis, pemfilteran, morfologi, deteksi fitur, dan banyak lagi. [5] Ini dirancang untuk beroperasi dengan pustaka numerik dan ilmiah Python NumPy dan SciPy.

Proyek scikit-image dimulai sebagai scikits.image, oleh Stéfan van der Walt. Namanya berasal dari gagasan bahwa ini adalah "SciKit" (SciPy Toolkit), ekstensi pihak ketiga yang dikembangkan dan didistribusikan secara terpisah untuk SciPy. [6] Basis kode asli kemudian secara ekstensif ditulis ulang oleh pengembang lain. Dari berbagai scikits, scikit-image dan scikit-learn digambarkan sebagai "terpelihara dengan baik dan populer" pada November 2012. [7] Scikit-image juga telah aktif di Google Summer of Code. [8]

* + 1. ***Numpy***

NumPy adalah *library* untuk bahasa pemrograman Python, menambahkan dukungan untuk *array* dan matriks multi-dimensi yang besar, bersama dengan banyak koleksi fungsi matematika tingkat tinggi untuk beroperasi pada array ini. [9] NumPy awalnya diciptakan oleh Jim Hugunin dengan kontribusi dari beberapa developer lain. Pada tahun 2005, Travis Oliphant membuat NumPy dengan memasukkan fitur Numarray yang bersaing ke dalam Numeric, dengan modifikasi ekstensif. NumPy adalah perangkat lunak sumber terbuka dan memiliki banyak kontributor.

* + 1. ***Scipy***

SciPy adalah *library* Python *open source* dan gratis yang digunakan untuk komputasi ilmiah dan komputasi teknis. [10] SciPy berisi modul untuk pengoptimalan, aljabar linier, integrasi, interpolasi, fungsi khusus, FFT, pemrosesan sinyal dan gambar, pemecah ODE, dan tugas lain yang umum dalam sains dan teknik.

SciPy dibangun di atas objek array NumPy dan merupakan bagian dari tumpukan NumPy yang mencakup alat-alat seperti Matplotlib, pandas dan SymPy, dan satu set perpustakaan komputasi ilmiah yang berkembang. Tumpukan NumPy ini memiliki pengguna yang mirip dengan aplikasi lain seperti MATLAB, GNU Octave, dan Scilab. Tumpukan NumPy juga kadang-kadang disebut sebagai tumpukan SciPy. [11]

SciPy juga merupakan keluarga konferensi untuk pengguna dan pengembang alat ini: SciPy (di Amerika Serikat), EuroSciPy (di Eropa) dan SciPy.in (di India). [12] Enthought berasal dari konferensi SciPy di ​​Amerika Serikat dan terus mensponsori banyak konferensi internasional serta menjadi tuan rumah situs web SciPy.

Perpustakaan SciPy saat ini didistribusikan di bawah lisensi BSD, dan pengembangannya disponsori dan didukung oleh komunitas pengembang terbuka. Ini juga didukung oleh NumFOCUS, sebuah yayasan komunitas untuk mendukung sains yang dapat direproduksi dan diakses.

* + 1. ***Matplotlib***

Matplotlib adalah pustaka plot untuk bahasa pemrograman Python dan ekstensi matematika numeriknya NumPy. Ini menyediakan API berorientasi objek untuk menanamkan plot ke dalam aplikasi menggunakan toolkit GUI tujuan umum seperti Tkinter, wxPython, Qt, atau GTK +. Ada juga antarmuka "pylab" prosedural berdasarkan mesin negara (seperti OpenGL), yang dirancang sangat mirip dengan MATLAB, meskipun penggunaannya tidak disarankan. [13] SciPy memanfaatkan Matplotlib.

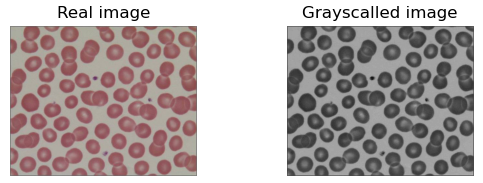
Matplotlib aslinya ditulis oleh John D. Hunter, sejak saat itu Matplotlib memiliki komunitas pengembangan yang aktif, [14] dan didistribusikan di bawah lisensi bergaya BSD. Michael Droettboom dinominasikan sebagai pengembang utama matplotlib tak lama sebelum kematian John Hunter pada Agustus 2012, [15] dan selanjutnya bergabung dengan Thomas Caswell. [16] [17]

Matplotlib 2.0.x mendukung Python versi 2.7 hingga 3.6. Dukungan Python 3 dimulai dengan Matplotlib 1.2. Matplotlib 1.4 adalah versi terakhir yang mendukung Python 2.6. [18] Matplotlib telah berjanji untuk tidak mendukung Python 2 setelah tahun 2020 dengan menandatangani Pernyataan Python 3. [19]

**BAB II**

**HASIL DAN PEMBAHASAN**

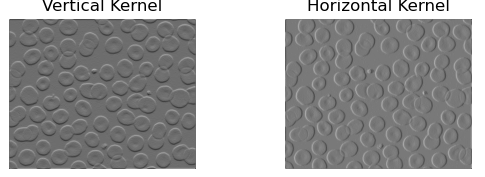
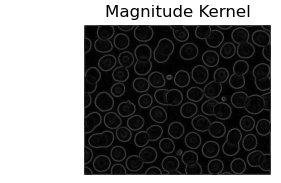
* 1. ***Loading* citra** **mikroskopik sel darah merah**



Gambar 3.1 Citra mikroskopik sel darah merah asli dan *grayscale*

Gambar 3.1 di atas merupakan hasil yang ditampilkan setelah melakukan proses pemuatan gambar ke dalam aplikasi *pycharm*. Terdapat 2 buah gambar, yaitu gambar asli dan gambar yang sudah diubah menjadi *grayscale.* Tujuan dilakukannya proses *grayscale* adalah untuk melakukan proses *edge detection.*

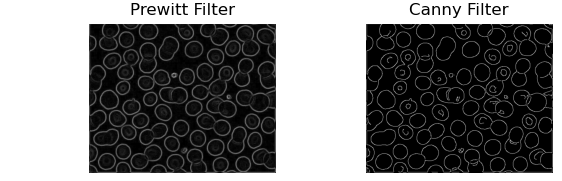
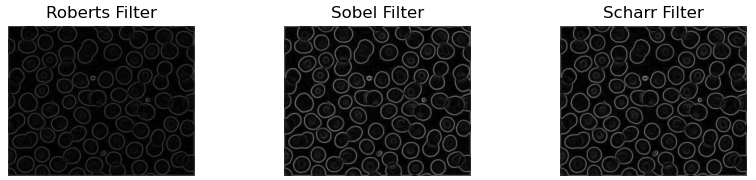
* 1. ***Edge detection* menggunakan *vertical* dan *horizontal kernel***

**** ****

Gambar 3.2 hasil proses *edge detection* menggunakan *vertical* dan *horizontal kernel*

Gambar 3.2 di atas merupakan gambar hasil proses *edge detection* menggunakan *vertical* dan *horizontal kernel.* Terlihat pada gambar *vertical kernel* merupakan proses mencari tepi dari sebuah objek dengan cara memindai citra tersebut pada jalur vertikal dari citra. Sedangkan gambar *horizontal kernel* merupakan proses mencari tepi dari sebuah objek dengan cara memindai citra tersebut pada jalur horizontal dari citra. Setelah didapatkan 2 buah jenis pendeteksian tepi di atas, Langkah selanjutnya adalah mencari *magnitude* dari kedua pendeteksian itu, sehingga didapatkan hasil tepi yang ingin dicari. Untuk hasil dari proses pencarian magnitud dapat dilihat pada gambar *magnitude kernel.*

* 1. ***Edge detection* menggunakan *library Scikit Image***

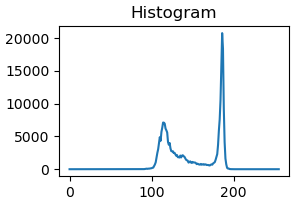
****

Gambar 3.3 hasil *Edge detection* menggunakan *library Scikit Image*

Gambar 3.3 di atas merupakan gambar citra sel darah merah yang sudah dilakukan proses *edge detection* menggunakan filter dari *library Scikit Image.* Terdapat beberapa jenis filter yang yang ditampilkan pada gambar tersebut, diantaranya filter Roberts, Sobel, Scharr, Prewitt, dan Canny. Untuk hasil citra yang didapatkan, filter dapat dikatakan berhasil melakukan proses pendeteksian tepi untuk setiap objek dalam citra tersebut, namun terdapat beberapa filter yang terlalu sensitif karena bagian dalam objek juga dideteksi sebagai objek lain, khususnya pada filter Canny. Untuk hasil yang sesuai dengan yang diinginkan dapat terlihat pada filter Roberts. Karena meskipun tepi yang dideteksi tidak terlalu cerah, namun filter berhasil mengurangi intensitas pada tepi dalam objeknya.

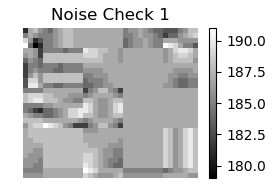
* 1. ***Edge detection* menggunakan *threshold***

Setelah dilakukan pemuatan dan proses *grayscale* pada citra, dilakukan proses menampilkan histogram untuk melihat sebaran intensitas dari citra yang akan diproses. Histogram dari citra *grayscale* dapat dilihat pada gambar 3.4. Pada gambar tersebut dapat terlihat bahwa terdapat 2 buah puncak untuk persebaran nilai *grayscale-*nya*.* Dari histogram ini juga harusnya sudah bisa dilakukan pengambilan *threshold* yang berada di antara kedua puncak tersebut. Namun, pada tugas ini penulis tetap mencoba mencari *threshold* menggunakan fungsi *otsu* sehingga dapat dibandingkan hasilnya antara *threshold* menggunakan *otsu* dengan yang menggunakan subjektifitas penulis.



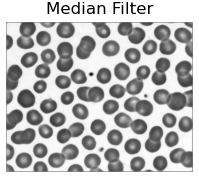
Gambar 3.4 histrogram untuk citra *grayscale*

Setelah didapatkan histogram dari citra yang sudah dilakukan proses *grayscale,* dilakukan proses pengecekan *noise* dari citra asli. Hasil pengecekan ini dapat dilihat pada gambar 3.5. Pada gambar tersebut, terlihat bahawa masih terdapat *noise* yang berada pada *background* dari citra yang mana ketika dicek untuk standar deviasinya, didapatkan hasil sebesar 1.443231446027532, dan dapat dikatakan nilai persebaran *noise-*nya masih tinggi. Untuk itu, dilakukan proses selanjutnya yaitu pemfilteran menggunakan *median filter.*

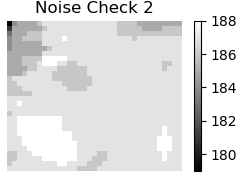


Gambar 3.5 pengecekan *noise* pada *background* dari citra

Hasil dari proses median filter dapat terlihat pada gambar 3.6. Dapat terlihat pada gambar tersebut bahwa *noise* yang berada pada *background* dari citra tereduksi dan *background-*nya memiliki intensitas yang lebih cerah dibandingkan sebelum melakukan proses *median filter.* Selanjutnya dilakukan proses untuk pengecekan nilai standar deviasi dari citra yang sudah dilakukan proses *median filter.* Hasil standar deviasinya berada pada nilai 0.7530340745678892 dan dapat dikatakan *noise* pada *background* berhasil dihilangkan yang kemudian ketika dilakukan proses *noise check* menghasilkan hasil seperti pada gambar 3.7.

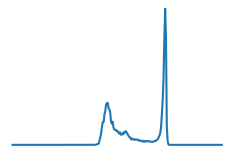


Gambar 3.6 hasil setelah dilakukan proses *median filter*



Gambar 3.7 pengecekan *noise* pada *background* dari citra setelah dilakukan proses *median filter*

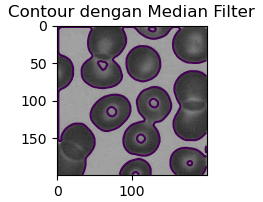
Kemudian untuk mengetahui persebaran intensitas dari citra setelah dilakukan proses *median filter,* ditampilkan juga histogram dari citra setelah dilakukan proses *median filter.* Hasilnya dapat dilihat pada gambar 3.8.



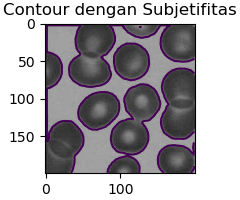
Gambar 3.8 histogramdari citra setelah dilakukan proses *median filter*

Ketika dilihat sekilas, memang tidak terdapat perbedaan antara histogram dari citra sebelum dan setelah dilakukan proses *median filter.* Namun, ketika dicermati lagi, puncak yang berada di sebelah kanan pada histogram dari citra setelah dilakukan proses *median filter* lebih rapat dibandingkan dengan histogram dari citra sebelum dilakukan proses *median filter.*

Setelah itu, dilakukan proses mencari *threshold* menggunakan citra setelah dilakukan proses *median filter.* Sehingga didapatkan nilai *threshold-*nyasebesar 152 yang mana ketika dilakukan proses *contouring* menggunakan nilai tersebut didapatkan hasil seperti pada gambar 3.9. Pada gambar tersebut terlihat bahwa proses kontur berhasil dilakukan, namun objek-objek yang berada dalam objek besar juga terdeteksi sebagai objek lain, sehingga menurut penulis ini masih belum sesuai dengan keinginan penulis, dimana kontur yang diinginkan adalah hanya berada pada tepi luar objek besar saja. Untuk itu penulis mencoba melakukan proses konturing menggunakan *threshold*  dari subjektifitas penulis dengan nilainya yaitu 170. Hasil dari proses konturing menggunakan nilai 170 ini dpat dilihat pada gambar 3.10

**

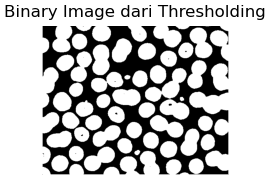
Gambar 3.9 *contouring* menggunakan *threshold* dari *median filter*



Gambar 3.10 *contouring* menggunakan *threshold* dari subjektifitas penulis dengan nilai 170

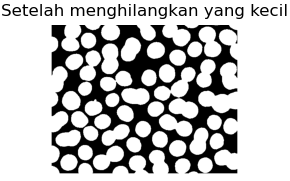
Pada gambar tersebut terlihat bahwa proses konturing yang dilakukan berhasil mendeteksi bagian tepi dari objek-objek besar saja, berbeda dari percobaan sebelumnya yang mana objek yang berada di dalam objek besar juga dideteksi sebagai objek lain.

Selanjutnya, karena proses konturing berhasil dilakukan, selanjutnya dilakukan proses *image masking* untuk benar-benar memisahkan antara objek dengan *background.* Hasil dari *image masking* dapat terlihat pada gambar 3.11.



Gambar 3.11 hasil *image masking*

Pada gambar tersebut telihat bahwa objek dan *background* yang terlihat berupa warna hitam dan putih saja yang mana warna putih merupakan objek, sedangkan warna hitam adalah *background.* Namun, apabila dicermati lagi, pada gambar tersebut masih terdapat objek-objek kecil yang bukan merupakan objek yang diinginkan oleh penulis, sehingga perlu dilakukan proses untuk menghilangkan objek-objek tersebut. Untuk hasil dari penghilangan objek-objek kecil tersebut dapat dilihat pada gambar 3.12.

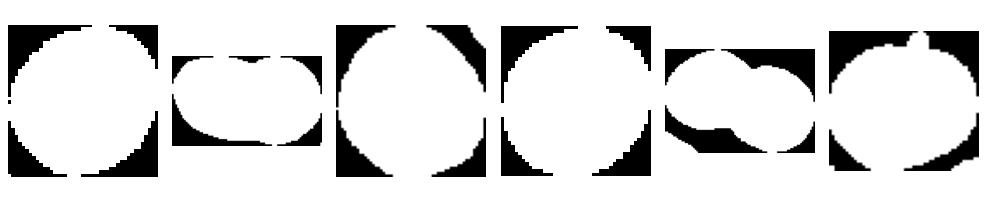


Gambar 3.12 hasil setelah dilakukan proses penghilangan objek-objek kecil

Pada gembar tersebut terlihat bahwa objek-objek kecil yang tidak diinginkan berhasil dihilangkan sehingga menghasilkan objek-objek besar saja.

* 1. **Menampilkan objek urutan 15 sampai 21**

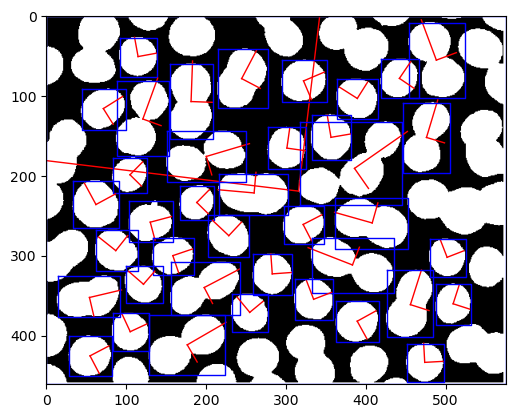
Setelah dilakukan proses *image masking,* kita dapat melakukan proses untuk menghitung jumlah objek yang terdeteksi. Untuk objek yang terdeteksi adalah sebanyak 40 objek. Dan berikut merupakan gambar untuk objek uritan ke-15 sampai 21.

****

Gambar 3.12 objek urutan 15 sampai 21

* 1. ***Image Labeling***

Hasil dari proses *image labeling* dapat dilihat pada gambar 3.13*.* Pada gambar terlihat bahwa terdapat objek-objek yang dideteksi sebagai objek yang sempurna, dimana objek yang sempurna ini merupakan objek yang tidak terpotong pada bagian tepi dari citra. Selain itu, objek-objek yang sempurna ini juga dilabelkan dengan kotak berwarna biru pada gambar tersebut. Kemudian dilakukan juga proses untuk mencari ‘*centroid', 'orientation', 'major\_axis\_length',* dan *'minor\_axis\_length'.*

****

Gambar 3.13 hasil setelah dilakukan *image labeling*

* 1. **Meng-*import Nilai ‘centroid', 'orientation', 'major\_axis\_length',* dan *'minor\_axis\_length'* yang didapatkan ke *Microsoft Excel***

Setelah didapatka data berupa *centroid*, *orientation, major\_axis\_length, dan minor\_axis\_length*, data tersebut dapat di-*import* ke dalam bentuk *Microsoft excel* dengan menggunakan fungsi “*excel.to\_excel”.* Pada tugas ini, penulis menyimpan *excel-*nyadengan nama "Exported\_Dataframe.xlsx". Untuk hasil pada *Microsoft excel-*nya dapat dilihat pada Tabel 3.1 di bawah

Table 3.1 nilai *centroid*, *orientation, major\_axis\_length, dan minor\_axis\_length* pada *Microsoft excel*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **centroid-0** | **centroid-1** | **orientation** | **major\_axis\_length** | **minor\_axis\_length** |
| **0** | 218 | 315 | 1,450535 | 830,1079 | 652,756 |
| **1** | 54 | 488 | 0,357238 | 107,667 | 53,4469 |
| **2** | 50 | 114 | 0,181452 | 48,87279 | 47,38006 |
| **3** | 77 | 244 | -0,4834 | 79,64088 | 52,55965 |
| **4** | 77 | 442 | -0,59567 | 50,28106 | 44,19404 |
| **5** | 80 | 321 | -1,16925 | 56,73789 | 50,64584 |
| **6** | 106 | 181 | -0,0368 | 101,5892 | 53,01565 |
| **7** | 102 | 389 | 1,006984 | 52,3242 | 48,49475 |
| **8** | 128 | 120 | -0,34846 | 108,9617 | 48,54446 |
| **9** | 115 | 71 | -1,01071 | 57,14134 | 46,26493 |
| **10** | 151 | 475 | -0,29235 | 97,9085 | 47,1227 |
| **11** | 151 | 356 | 0,168813 | 56,40749 | 50,52852 |
| **12** | 190 | 385 | -0,96156 | 161,4897 | 61,24751 |
| **13** | 165 | 301 | -0,1401 | 51,46236 | 46,88048 |
| **14** | 175 | 200 | -1,28188 | 112,7939 | 48,16727 |
| **15** | 198 | 104 | -0,77811 | 45,27223 | 40,29498 |
| **16** | 221 | 260 | 1,481707 | 90,53129 | 51,78585 |
| **17** | 235 | 62 | 0,496091 | 60,01064 | 55,67518 |
| **18** | 233 | 188 | -0,77571 | 45,18144 | 40,68162 |
| **19** | 258 | 408 | 1,295592 | 97,91467 | 52,15332 |
| **20** | 257 | 130 | -1,31077 | 54,83771 | 46,55011 |
| **21** | 260 | 321 | -1,09373 | 52,56341 | 45,75497 |
| **22** | 274 | 228 | 0,816833 | 55,90743 | 46,51216 |
| **23** | 293 | 86 | 0,899422 | 55,10427 | 48,04451 |
| **24** | 299 | 158 | -1,24425 | 52,00254 | 45,29508 |
| **25** | 311 | 383 | 1,198667 | 119,262 | 47,7705 |
| **26** | 301 | 501 | 0,370887 | 47,28675 | 44,81625 |
| **27** | 322 | 282 | 0,071851 | 50,11145 | 48,15487 |
| **28** | 339 | 197 | -1,08898 | 101,0424 | 45,09813 |
| **29** | 335 | 121 | 0,855916 | 49,1184 | 43,18696 |
| **30** | 361 | 455 | -0,30768 | 90,89048 | 48,92954 |
| **31** | 352 | 53 | -1,33976 | 82,34691 | 48,47354 |
| **32** | 354 | 334 | 0,366223 | 51,28815 | 48,3615 |
| **33** | 359 | 509 | -0,31205 | 50,35579 | 43,2357 |
| **34** | 370 | 254 | 0,691021 | 49,7938 | 43,48131 |
| **35** | 381 | 389 | -1,06666 | 53,87525 | 50,45845 |
| **36** | 394 | 104 | 0,436165 | 47,82292 | 45,75736 |
| **37** | 411 | 176 | -1,04588 | 112,3703 | 49,7108 |
| **38** | 425 | 54 | -1,08804 | 53,48638 | 49,77122 |
| **39** | 433 | 473 | 0,054487 | 47,71738 | 46,17961 |

Pada tabel tersebut terlihat data berjumlah 40 buah yang memiliki penomoran dari 0 sampai 39.

**BAB III**

**KESIMPULAN**

Kesimpulan yang dapat diambil dari tugas ini adalah :

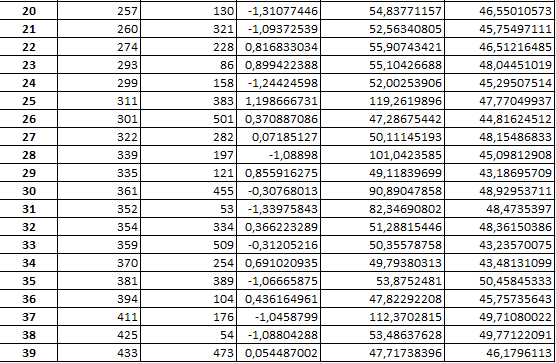
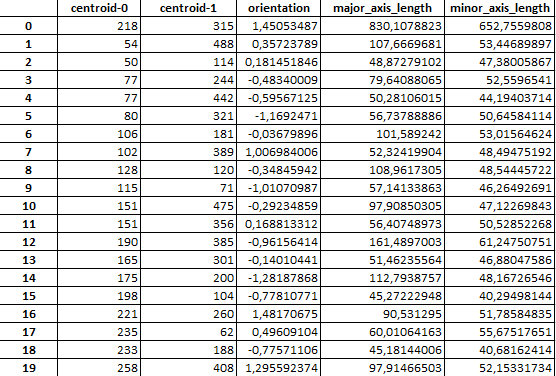
1. Sebelum melakukan proses apapun pada sebuah citra, perlu dilakukan proses untuk mengubah citra asli menjadi *grayscale.*
2. Proses *edge detection* menggunakan metode *magnitude kernel* dilakukan dengan mencari magnitudo dari *vertical kernel* dan *horizontal kernel.*
3. Filter dari *library Scikit image* terbaik untuk mendeteksi tepi pada citra mikroskopik sel darah merah adalah filter *Roberts.*
4. Proses *edge detection* menggunakan *thresholding otsu* tidak selamanya sesuai untuk mendeteksi bagian tepi dari objek yang diinginkan.
5. Terdapat 40 objek yang terdeteksi dari citra mikroskopik sel darah merah pada tugas ini yang merupakan objek sempurna (tidak terpotong).
6. Informasi berupa *centroid*, *orientation, major\_axis\_length, dan minor\_axis\_length* dari objek yang dideteksi dapat diimport ke bentuk *Microsoft excel.*

**DAFTAR PUSTAKA**

1. ["A Python Book: Beginning Python, Advanced Python, and Python Exercises"](https://web.archive.org/web/20120623165941/http:/cutter.rexx.com/~dkuhlman/python_book_01.html). web.archive.org. 2012-06-23. Diakses tanggal 2020-08-11.
2. ["Welcome to Python.org"](https://www.python.org/about/). Python.org (dalam bahasa Inggris). Diakses tanggal 2020-08-11.
3. ["History and License — Python 3.8.5 documentation"](https://docs.python.org/3/license.html). docs.python.org. Diakses tanggal 2020-08-11.
4. S van der Walt; JL Schönberger; J Nunez-Iglesias; F Boulogne; JD Warner; N Yager; E Gouillart; T Yu; the scikit-image contributors (2014). "scikit-image: image processing in Python". PeerJ. 2:e453: e453. [*arXiv*](https://en.wikipedia.org/wiki/ArXiv_(identifier)):[*1407.6245*](https://arxiv.org/abs/1407.6245). [*doi*](https://en.wikipedia.org/wiki/Doi_(identifier)):[*10.7717/peerj.453*](https://doi.org/10.7717%2Fpeerj.453).
5. Chiang, Eric (2014). [*"Image Processing with scikit-image"*](http://blog.yhathq.com/posts/image-processing-with-scikit-image.html).
6. Dreijer, Janto. [*"scikit-image"*](https://scikits.appspot.com/scikit-image).
7. Eli Bressert (2012). [*SciPy and NumPy: an overview for developers*](https://books.google.com/books?id=fLKTuJqQLVEC&pg=PA43). O'Reilly. p. 43.
8. Birodkar, Vighnesh (2014). [*"GSOC 2014 – Signing Off"*](https://vcansimplify.wordpress.com/2014/08/17/gsoc-2014-signing-off/).
9. Charles R Harris; K. Jarrod Millman; Stéfan J. van der Walt; et al. (16 September 2020), [*"Array programming with NumPy"*](https://www.nature.com/articles/s41586-020-2649-2.pdf) *(PDF)*, [*Nature*](https://en.wikipedia.org/wiki/Nature_(journal)) (in English and English), **585**(7825): 357–362, [*doi*](https://en.wikipedia.org/wiki/Doi_(identifier)):[*10.1038/S41586-020-2649-2*](https://doi.org/10.1038%2FS41586-020-2649-2), [*ISSN*](https://en.wikipedia.org/wiki/ISSN_(identifier)) [*1476-4687*](https://www.worldcat.org/issn/1476-4687), [*PMID*](https://en.wikipedia.org/wiki/PMID_(identifier)) [*32939066*](https://pubmed.ncbi.nlm.nih.gov/32939066), [*Wikidata*](https://en.wikipedia.org/wiki/WDQ_(identifier)) [*Q99413970*](https://www.wikidata.org/wiki/Q99413970)
10. Pauli Virtanen; Ralf Gommers; Travis E. Oliphant; et al. (3 February 2020), "SciPy 1.0: fundamental algorithms for scientific computing in Python" (PDF), Nature Methods, 17 (3): 261–272, doi:10.1038/S41592-019-0686-2, ISSN 1548-7091, PMC 7056644, PMID 32015543, Wikidata Q84573952
11. "Scientific Computing Tools for Python". SciPy.org.
12. "SciPy Conferences".
13. "Matplotlib coding styles". matplotlib.org.
14. "Matplotlib github stats". matplotlib.org.
15. "Announcing Michael Droettboom as the lead Matplotlib developer". matplotlib.org.
16. "Matplotlib Lead Developer Explains Why He Can't Fix the Docs—But You Can – NumFOCUS". NumFOCUS. 2017-10-05. Retrieved 2018-04-11.
17. "Credits – Matplotlib 2.2.2 documentation". matplotlib.org. Retrieved 2018-04-11.
18. "Installing – Matplotlib 2.0.2 documentation". Retrieved 2017-06-23.
19. "Add Matplotlib to list by takluyver · Pull Request #20 · python3statement/python3statement.github.io". GitHub. Retrieved 2018-04-11.

**Lampiran**

* + 1. ***centroid, orientation, major\_axis\_length, dan minor\_axis\_length* pada *Microsoft excel***

****