Tesi

# Elaborato prodotto

## plugin in imagej

I componenti principali e di base di ImageJ sono i Plugin che estendono e arricchiscono il tool in maniera efficace. Ogni elemento, compresa l’interfaccia con cui l’utente interagisce con il tool, è creata con i Plugin. Un Plugin si sviluppa generalmente con il linguaggio di programmazione Java. Qualsiasi utente può iniziare la stesura di un Plugin personalizzato e volto ad un determinato obbiettivo; è possibile consultare la documentazione online, riferita alle classi utilizzate e sfruttate da ImageJ. Attraverso il menu principale, scegliendo la voce “Plugins”, si può iniziare la scrittura attraverso il comando “new” e scegliendo il Plugin base prescelto. Il particolare ImageJ ne offre quattro:

1. Plugin
2. Plugin Filter
3. Plugin Frame
4. Plugin Tool

Per concentrarsi sulla individuazione e calcolo delle misure richieste per l’elaborato i Plugin utilizzati sono della tipologia Plugin e Plugin Filter. Ogni tipologia corrisponde a una interfaccia implementata a seconda dello scopo del Plugin. Il Plugin prodotto ai fini dell’elaborato si serve delle due interfacce PlugIn e PlugInFilter. PlugIn riporta un solo metodo chiamato run che prende come argomento un oggetto di tipologia String. Il metodo viene chiamato quando viene avviato il Plugin. L’argomento di classe String può essere anche vuoto, ovvero “”. Plugin Filter è leggermente più corposo; al suo interno vi è il metodo run e il metodo setup. Quest’ultimo viene chiamato quando il Plugin viene caricato. Presenta due argomenti passati uno di tipologia String che può essere vuoto e un oggetto di classe ImagePlus che indica l’immagine attualmente attiva. Restituisce un valore di tipologia intero che indica se il metodo è andato a buon fine., medesimo nome della interfaccia citata sopra, ma con un obbiettivo diverso; utilizzano il metodo per elaborare l’immagine e presenta una serie di flag di controllo, pubbliche. Prende in ingresso un oggetto della classe ImageProcessor. Quest’ultima è volta ad applicare alcuni metodi base per manipolare immagini 2D.

## Plugin dell’elaborato

### Introduzione

Nonostante ImageJ sia ricco di Plugin per poter trattare e calcolare molte features data in ingresso una immagine, nessun Plugin prende tutte le misure necessarie per raggiungere obbiettivo proposto dell’elaborato, o quelle presenti non sono raggruppate nello stesso Plugin. L’idea sarebbe poter sfruttare al meglio i Plugin, dunque le classi, esistenti su cui si basa ImageJ e estrarre le misure richieste relative ad immagini biomediche. In particolare, il Plugin che si avvicina di più a queste esigenze è “ParticlesAnalyzer”, avviabile dal menu “Analyze” con la voce “Analyze Particles…”. L’elemento più importante di ParticlesAnalyzer è la possibilità di individuare più oggetti o regioni di interesse nella immagine e calcolare le features su ognuno di essi, separatamente. Questo avviene attraverso un Threshould manuale, con cui si attua la definizione delle aree di interesse attraverso il plugin “Threshould” accessibile dal menu sotto le voci “Image”, “Adjust” e “Threshould”. Se si tratta di immagini in RGB, ovvero a colori, si usa invece “Color Threshould” accessibile da medesimo menu. ParticlesAnalyzer implementa PlugInFilter e Measurements. Measurements è una interfaccia che lo collega ad un menu in cui si possono settare alcune misure ma ai fini dell’elaborato non sarà rilevante, poiché sarà sostituito da una seconda interfaccia in cui si possono selezionare le misure implementate e esistenti. Oltre alla possibilità di analizzare più regioni di interesse nella immagine, ParticlesAnalyzer offre una elaborazione postuma ai calcoli effettuati, per visualizzare le regioni di interesse in maniera differente rispetto all’immagine originale, per esempio evidenziando solo i contorni e inserendo il numero della regione al suo interno. Date le proprietà di questo Plugin la decisione presa per arricchire e sfruttare le features implementate precedentemente è quella di creare un nuovo Plugin che estendesse Particles Analyzer e basarsi sul suo funzionamento.

### Sviluppo e storia del plugin

Prima di iniziare la scrittura del codice riguardante il Plugin prodotto per l’elaborato, si è avviato un periodo di ricerca e di incremento delle conoscenze sul tool da utilizzare ovvero ImageJ; sono stati presi in considerazione anche altri Plugin che simulavano lo scopo dell’elaborato ma o le misure non coincidevano interamente con le misure richieste oppure lo stile non era conforme con le aspettative del prodotto finale. Il Plugin prodotto è stato aggiornato volta per volta basandosi su tre fasi; il primo stadio prevedeva la implementazione di features riguardanti immagini in bianco e nero, tipologia di immagine per cui i pixels che la compongono sono o bianchi o neri; il secondo stadio prevedeva la implementazione delle misure per immagini in scala di grigio; il terzo stadio e ultimo prevedeva la implementazione di features per immagini a colori, ovvero in RGB. Durante la stesura del codice e la ricerca delle metodologie da applicare per le formule, per la maggior parte note, all’interno del Plugin, si è voluto curare anche l’interfaccia in modo tale da poter scegliere se implementare tutte le misure proposte, solo alcune oppure combinarle. Inoltre la struttura del Plugin si è evoluta continuamente grazie alle metodologie scoperte e applicate durante la stesura dell’elaborato. La particolarità dei Plugin di ImageJ è la necessità di appoggiarsi completamente sulle classi appartenenti alla libreria dedicata al tool, dunque anche il Plugin prodotto ha seguito questa ideologia.

### Classi principali utilizzate

Per comprendere la struttura e il funzionamento dell’elaborato, è necessario conoscere le classi di base utilizzate più frequentemente. Per classi si intendono Plugin creati appositamente per ImageJ volte al trattamento delle immagini in ingresso. Alcune classi utilizzate sono associate ad altre e collaborano con esse, con l’obbiettivo di elaborare o estrapolare fattori e elementi che caratterizzano l’immagine. La classe che prende in ingresso l’immagine, porta il nome di IJ ed è fondamentale per il funzionamento del tool stesso. Attraverso il metodo statico IJ.getImage(), viene restituito un oggetto appartenente alla classe ImagePlus con cui l’immagine viene analizzata. Un oggetto ImagePlus contiene un oggetto della classe astratta ImageProcessor, super classe di ByteProcesor, ColorProcessor, FloatProcessor, ShortProcessor. Questi ultimi sono associati alle tipologie di dati, byte, short, float e RGB, supportati da ImageJ. Un ImageProcessor contiene i dati dei pixel delle immagini in 2D e i metodi base per manipolarli. Un oggetto appartenente alla classe ImagePlus può contenere un oggetto della classe ImageStack, volto per immagini 2D, 3D, 4D o 5D. Nel caso specifico dell’elaborato verranno trattate immagini in 2D. Viene utilizzata all’interno del Plugin per poter trattare su più elaborazioni e trasformazioni dell’immagine originale. Per poter individuare le zone di interesse viene utilizzata la classe ROI, inteso come “region of interest”, regione di interesse. Un oggetto appartenente a tale classe può essere creato attraverso più costruttori; sono accumunati dall’utilizzo delle soglie decise attraverso il Threshould. La classe ROI si serve della classe Polygon e della sotto classe Rectangle per individuare i punti e le coordinate da cui è composto e per l’individuazione del suo Convex Hull. Per poter calcolare dati statistici e morfologiche si serve di un oggetto della classe ImageStatistics. Ogni ROI ha associata un suo oggetto ImageStatistics il quale viene creato attraverso il metodo getStatistics(). Tramite tale oggetto si può anche accedere all’istogramma che caratterizza la regione selezionata. L’istogramma si presenta come vettore per cui gli indici simboleggiano il livello di grigio e il dato intero associato il numero di pixel che detengono quel livello in particolare. Per poter ricavare features da un immagine in ingresso RGB è stata utilizzato un oggetto appartenente alla classe ChannelSplitter, per poter ricavarne un array di ImagePlus dati i suoi canali, attraverso il metodo spli(ImagePlus imp). Per quanto riguarda l’interfaccia, viene sfruttata la classe GenericDialog, volta a visualizzare una finestra di dialogo a cui si possono aggiungere messaggi o check box, aree di testo, menu a tendina e altre inserzioni grafiche. Per visualizzare i risultati ottenuti si utilizza la classe ResultsTable che visualizza una tabella contenente i risultati ottenuti.

### Struttura del Plugin

#### Catch\_Parasite3

“Catch\_Parasite3” (?) è il nome attuale del Plugin volto all’obbiettivo proposto dall’elaborato. Esso implementa l’interfaccia PlugIn e al suo interno vi sono due metodi, run e getImagePlusHSB, e una inner class Parasite. Gli attributi globali dichiarati sono imp, appartenente alla classe ImagePlus e deposito dell’immagine in ingresso, impPlusesRGB, impPlusesHSB, vettori appartenenti alla classe ImagePlus e un attributo booleano typeRGB. Questi ultimi tre sono volti al trattamento delle immagini a colori. Il metodo run viene avviato all’esecuzione del Plugin per settare e avviare la inner class Parasite, volta al trattamento e al calcolo delle misure proposte dall’elaborato. Al suo interno, viene inizializzata la variabile globale imp che riceve l’immagine in ingresso, con il metodo di IJ getImage(). Viene attuata la verifica della tipologia di immagine con l’esecuzione del metodo getType richiamato dall’oggetto imp. Se questo rende la variabile statica intera COLOR\_RGB l’immagine è a colori e si può procedere con la inizializzazione dei vettori dedicati impPlusesRGB e impPlusesHSB. Le misure per immagini a colori sono state fatte sfruttando la proprietà per cui una immagine in grigio è uguale alla somma dei valori dei pixel in rosso, in verde e in blu diviso tre oppure tramite la formula proposta da ImageJ stesso:

Di conseguenza si può trattare l’immagine e estrapolarne la sua versione grigia ma considerando un canale per volta ovvero corrispondente al rosso, verde e blu. Le immagini ottenute saranno trattate come immagini in scala di grigio a cui si possono applicare le stesse formule. Per ottenerle viene utilizzato un oggetto della classe ChannelSplitter, su cui viene richiamato il metodo split che prende in ingresso l’oggetto imp. Dopo aver ottenuto l’inizializzazione del primo vettore impPlusesRGB si procede come l’inizializzazione di impPlusesHSB con il metodo getImagePlusHSB. L’immagine verrà trattata in HSB ovvero tonalità, saturazione e luminosità; esso è un metodo additivo di composizione dei colori. Dai tre parametri, la tonalità varia partendo convenzionalmente dal rosso primario a 0°, passando per il verde primario a 120° e il blu primario a 240° e quindi tornando al rosso a 360°; per saturazione si intende l’intensità e la purezza della singola tonalità mentre la luminosità e un’indicazione della sua brillantezza. Prendendo singolarmente tali fattori, si ottengono immagini in scala di grigio differenti dalla originale o dalle corrispondenti nei canali RGB. Il metodo dedicato, getImagePlusHSB() che prende come parametro l’ImagePlus, al suo interno vengono dichiarati gli ImageStack corrispondenti a Hue, Saturation e Brightness. L’immagine sotto forma di oggetto ImagePlus viene salvata in un quarto ImageStack e da esso si ricava un oggetto di classe ColorProcessor. Quest’ultimo è una sotto classe di ImageProcessor, il quale contiene i dati dei pixel. Con l’utilizzo del ColorProcessor si ottengono i vettori corrispondenti ai dati dei pixel trasformati singolarmente in HSB per poi settare gli ImageStack sotto forma di byte. Dagli ImageStack vengono creati gli ImagePlus corrispondenti alle immagini in Hue, Saturation e Brightness e assegnati al vettore; alla posizione zero l’immagine trattata per tonalità alla posizione uno l’immagine trattata per saturazione alla posizione due l’immagine trattata per luminosità. Il vettore, così composto, viene restituito al vettore impPlusesHSB. Dopo aver terminato le operazioni necessarie per il trattamento delle immagini a colori viene creato un oggetto parasite, ovvero appartenente alla inner class Parasite del Plugin stesso. Avviando il setup se è andato a buon fine la variabile intera restituita è uguale all’attributo statico DONE di PlugInFilter. In questo caso si può procedere altrimenti il Plugin blocca il suo funzionamento. Solitamente questo avviene se sono state chiuse le finestre di interfaccia iniziali o se non sono state trattate le immagini con il Threshould per definire le regioni di interesse. Se l’esecuzione non viene interrotta viene richiamato il metodo run dell’oggetto parasite. Alla fine della sua esecuzione verranno visualizzati i risultati in una ResultsTable.

#### Parasite

Parasite è la inner class del Plugin principale, ovvero di Catch\_Parasite3. La sua implementazione era necessaria perché estendendo la classe ParticlesAnalyzer bisognava avere un appoggio per far avviare il Plugin, in particolare i metodi setup e run. Parasite si occupa di implementare e raggruppare le misure richieste al fine di raggiungere l’obbiettivo dell’elaborato e sfruttare al meglio la sua super classe. All’interno della classe Parasite vi sono gli attributi che lo caratterizzano e che sono necessari per settare per features scelte. L’attributo intero measure, necessario per settare le misure presenti in ImageJ. ImageJ si serve della interfaccia Measurements per segnalare le features da calcolare. Solitamente la scelta di tali misure è attuata attraverso il menu SetMeasurements, accessibile dal comando del menu principale Analyzer. Questo menu è necessario per ParticlesAnalyzer per visualizzare solo le misure scelte. Nel caso dell’elaborato descritto non serve dover settare le misure da tale menu poiché le features sono raggruppate in una unica finestra visualizzata dallo stesso Plugin, in particolare dalla inner class Parasite. A seguire, vi sono i vettori booleani corrispondenti alle features implementate suddivisi per tipologia di immagine, bianco e nerp, scala di grigi e a colori con i rispettivi attributi booleani indicanti la misura implementata. Inizialmente vettori e attributi non sono connessi, verranno sfruttati per collegare le scelte proposte e selezionate nel menu di settaggio e le misure da visualizzare nella ResultTable. Seguono i metodi necessari per poter aggiungere le misure del Plugin e per impostare l’interfaccia; due di essi sono Override dei metodi della super classe. Il primo metodo è showDialog che non prende argomenti in ingresso e restituisce un valore di tipo intero. Esso è il primo Override e richiama il metodo della super classe. Questo visualizzerà un menu iniziale già implementato da ParticlesAnalyzer. Questo metodo si occupa anche di verificare se il setup è andato a buon fine. L’Override era necessario per richiamare una nuova finestra di dialogo per settare le misure da visualizzare. La finestra di dialogo in questione viene avviata con il metodo dialogSetMeasumenents(); solo se nella prima finestra di dialogo è stato selezionato il pulsante “Ok”, la seconda verrà visualizzata. All’interno di tale metodo vengono visualizzate le scelte attraverso check box e raggruppate a seconda della implementazione o della tipologia di features da presentare. In ordine si ha: features implementate da ImageJ, riprese dal menu SetMeasurements citato sopra, features implementate durate lo studio e la creazione dell’elaborato riguardanti le misure per immagini in bianco e nero, features riguardanti le immagini in grigio, features riguardanti le immagini a colori. Queste ultime non appariranno se l’immagine non è a colori. Allo stesso modo la misura “min and max” per le immagini in scala di grigio non vengono inserite nella finestra di dialogo se l’immagine è a colori poiché darebbe valore zero, poiché si ricerca il massimo e il minimo del livello di grigio presente nell’immagine. I valori selezionati vengono immessi nei vettori booleani di appoggio riguardanti le misure di ImageJ, le misure in Bianco e Nero, le misure in grigio e le misure in RGB. Ogni check box esistente, corrisponde ad un valore booleano, true o false. Per recuperare tali valori, vengono fatti scorrere e salvati nei vettori di appoggio per la selezione delle misure. Attraverso il metodo setMeasurementsExstended() vengono recuperati valori dai vettori di appoggio e, in base all’indice, viene assegnato il valore booleano agli attributi di appoggio corrispondenti alle misure implementate; per esempio “doConvexArea” se verrà settato come vero nella ResultsTable apparirà tale valore. L’Override più importante è del metodo saveResults. Viene prima richiamato il metodo della super classe per poi aggiungere le misure da visualizzare. Vengono presi in ingresso un oggetto della classe ROI e il suo oggetto stats della classe ImageStatistics. L’oggetto roi è l’area di interesse che occupa l’oggetto singolo analizzato in quel momento, a seconda della posizione nell’immagine. Sono necessarie alcuni attributi per il calcolo di determinate misure tra questi un oggetto della classe Polygon reso dall’oggetto roi attraverso il metodo getConvexHull(), per poter ottenere, come suggerisce il nome stesso, il suo convex hull. Grazie alla creazione di tale oggetto, vengono calcolati area e perimetro del convex hull, chiamando due metodi interni della classe, getArea e getPerimeter. Vengono anche salvati i dati relativi al Feret e del perimetro dell’oggetto e un vettore di interi popolato dall’istogramma, dato dall’oggetto ImageStatistics. Con una serie di strutture di controllo if consecutivi, viene verificata la necessità di calcolare le features attraverso i booleani settati precedentemente. Se l’immagine è in RGB si controllano anche le misure relative a quella tipologia. Dunque si crea un oggetto ImageStatistics dalle immagini ricavate e recuperate dai due vettori impPlusesRGB e impPlusesHSB, dopo averle settato la regione di interessa presa in considerazione, con il metodo setRoi(Roi roi). Sono stati implementati i metodi di appoggio per calcolare l’area e il perimetro del convexHull e il calcolo delle features Bending Energy, HeralickRatio, Entropy, InsentisitySum, Uniformity e Smothness. A seguire una serie di metodi di appoggio per la distanza, calcolo del logaritmo in base due e calcolo di operazioni per i vettori.

### Misure inserite nel Plugin ricavate da ImageJ

|  |  |
| --- | --- |
| Nome della misura | Descrizione |
| Area | Area della regione di interesse |
| Perimeter | Perimetro della regione di interesse |
| Centroid | Centroide o centro geometrico della regione, ovvero la posizione media di tutti i suoi punti inteso anche come media aritmetica della posizione di ciascuno di essi. Vengono visualizzati le coordinate x e y. |
| Center of mass | Centro della massa della regione, punto geometrico corrispondente al valore medio della distribuzione della massa del sistema nello spazio. Vengono visualizzati le coordinate xm e ym. |
| Bounding Rectangle | Valore dell’insieme dei punti del rettangolo della scatola di delimitazione (bounding box) della regione |
| Fit Ellipse | Restituzione dell’asse maggiore, minore e dell’angolo dell’ellissi adattata alla regione di interesse. |
| Shape descriptors | Calcolo di Circularity, AR, Roundness, e Solidity. |
| Feret’s diameter | Calcolo del Feret inteso come il massimo diametro della regione di interesse, dell’angolo del feret e del breadth inteso come il minimo diametro della regione di interesse. |
| Stack Position | Posizione della immagine nello stack di immagini |
| Mean | Media del valore grigio presente nella regione di interesse |
| Skewness | Grado di assimmetria della distribuzione dei valori di intesità del livello di grigio |
| Kurtosis K | La curtosi è un allontanamento della normalità distributiva, rispetto alla quale si verifica un maggiore appiattimento o un maggiore allungamento, applicata ai valori di grigio  Una misura del grado in cui la distribuzione dei valori di intensità di grigio si discosta dalla curva gaussiana |
| Median | La mediana dei valori dei pixels nell’area di interesse. |

### Misure implementate per le immagini in bianco nero

|  |  |
| --- | --- |
| Nome della misura | Descrizione |
| Convex Area | Area del poligono convesso.  Somma di tutte le distanze tra i; dato un punto A e un Punto A-1 e divisione per due. (Tratta il convex hull come se fosse composto da tanti triangolo?) |
| Convex Perimeter | Perimetro del poligono convesso  Somma delle distanze tra tutti i punti del contorno del Convex Hull |
| MinR (?) | Raggio del cerchio inscritto con punto di origine uguale al centro della massa interessata.  Divisione del minimo diametro per due |
| MaxR (?) | Raggio del cerchio inscritto con punto di origine uguale al centro della massa interessata  Divisione del massimo diametro, chiamato feret, per due |
| AspRatio | Chiamato anche Feret Ratio o Eccentricy o Rectangular Ratio. |
| Roundness |  |
| ArEquivD | Diametro del cerchio con l’equivalente area. |
| PerEquivD | Diametro del cerchio con l’equivalente perimetro. |
| EquivEllAr | Area dell’ellissi con il Feret come asse maggiore e il Breath asse minore. |
| Compactness |  |
| Concavity |  |
| Convexity | Chiamata anche rugosity o roughness. |
| Shape | Chiamata anche Thinness ratio. |
| RFactor |  |
| ArBBox |  |
| Rectangularity |  |
| Modification ratio | ( |
| Sphericity | Radius ratio. |
| Elongation | L’inverso di circularity. |
| Haralick Ratio | Calcolo della media dei raggi e della deviazione standard di essi, tramite metodo getHaralickRatio. |
| Bending Energy (?) |  |

### Misure implementate per le immagini in scala di grigio e per le immagini a colori

|  |  |
| --- | --- |
| Nome della misura | Descrizione |
| Entropy | Misura della casualità, tramite metodo getEntropy.  Data l’area e l’istogramma dei grigi, somma della probabilità che avvenga quel livello di grigio e lo stesso valore in logaritmo di due, per tutti i valori dell’istogramma, dunque fino a 256. Scarto del livello di grigio nel caso non ci siano pixel con tale gradazione. |
| Intensity | Somma delle intensità di livello di grigio presenti nell’oggetto, tramite metodo getIntensitySum.  Dato l’istogramma dei grigi, somma del livello per la sua quantità di pixel aventi quel livello e somma totale. |
| Square Intensity Sum | Radice quadrata di Intensity. |
| Uniformity | Una misura sulla uniformità, tramite metodo getUniformity.  Dato l’istogramma dei grigi e l’area calcolo della probabilità che il livello di grigio considerato si presenti nell’area. Tale probabilità viene elevata al quadrato. Vengono sommati tutti i risultati. |
| Smoothness R | Una misura sulla relativa levigatezza nella regione. Se R è uguale a 0 allora vi sono valori uguali in tutta la regione se invece si avvicina ad 1 oppure è uguale ad 1 la regione ha grandi escursioni di valori. Tramite metodo getSmoothness.  Data la varianza si calcola la varianza normalizzata e si esegue la seguente formula: |
| Min and Max | Il minimo e il massimo della intensità di grigio presente nella regione, nelle immagini a colori non è possibile calcolarlo. |

|  |  |
| --- | --- |
| Average red color, R mean | Media dei valori in rosso |
| Average green color, G mean | Media dei valori in verde |
| Average blue color, B mean | Media dei valori in blu |
| Red color std deviation, R std | Deviazione standard dei valori in rosso |
| Green color std deviation, G std | Deviazione standard dei valori in verde |
| Blue color std deviation, B std | Deviazione standard dei valori in blu |
| Square root of the value mean of channel R, G, B | Radice quadra della media del rosso, del verde e del blu. |
| Avarage RGB | La media RGB |

|  |  |
| --- | --- |
| Average Hue color, H mean | Media dei valori in Hue |
| Average Saturation color, S mean | Media dei valori in Saturation |
| Average Brightness color, B mean | Media dei valori in Brightness |
| Hue color std deviation | Deviazione standard dei valori in Hue |
| Saturation color std deviation | Deviazione standard dei valori in Saturation |
| Intensity color std deviation | Deviazione standard dei valori in Brightness |

### Utilizzo del Plugin

Il Plugin dell’elaborato ha come scopo semplificare il più possibile il calcolo immediato delle misure riguardanti immagini biomediche. Per poter iniziare, bisogna settare l’immagine manualmente e individuarne le soglie. Nel caso di immagini bianche e nere e in grigio si procede accedendo al comando di ImageJ Threshold. Si accede attraverso la combinazione di tasti Ctrl+Maiusc+T oppure dal menu stesso, selezionando Image>>Adjust >>Threshold. Per poter definire le soglie, si spostano i cursori e si settano a proprio piacimento le soglie e dunque le regioni di interesse come nell’esempio. Una volta decisi si esce tramite comando “X”. “Apply” rende l’immagine bianco e nera e non conserva le soglie indicate per le immagini in scala di grigio. Per quanto riguarda la definizione delle soglie in una immagine a colori si sceglie dallo stesso menù “Color Threshold”. Qui è possibile settare attraverso i valori di Hue, Saturation e Brightness, sempre spostando i cursori e poter scegliere altre opzioni a proprio piacimento. Terminato il settaggio si seleziona il comando “Select”. Una volta fatte queste operazioni preliminari, è possibile avviare il Plugin dal menu Plugins>>Catch\_Pariste3. Il primo menu che si presenta è possibile scegliere la grandezza minima dell’oggetto analizzato e a seguire quella massima e la sua circolarità. Attraverso il menu a tendina si può scegliere il come visualizzare l’immagine al termine della esecuzione del Plugin. Le scelte più utilizzate durante la stesura dell’elaborato sono: “nothing”, per cui l’immagine non viene trattata, “outlines” per cui l’immagine viene trattata e come risultato si ha una seconda immagina bianca dove si vedono solo ed esclusivamente i contorni degli oggetti con una etichetta che sta ad indicare il numero dell’oggetto, ”bare outlines” per cui si vedono solo i contorni degli oggetti, ellipses per cui l’immagine risultante è a sfondo bianco con delle “ellipsi” orientate al posto degli oggetti individuati, “mask” che restituisce l’immagine della maschera in bianco e nero, nero per gli oggetti e bianco lo sfondo, “overlay” che riprendendo l’immagine originale etichetta gli oggetti e infine “overlay masks” che riprende l’immagine originale e nelle zone dove si trova la maschera individuata sostituisce con un colore celeste chiaro. Le altre opzioni interessanti da settare attraverso check box sono “display results” per visualizzare i risultati, “clear results” per cancellare i risultati precedenti. Selezionato “Ok” si apre un secondo menu in cui si possono selezionare le misure. Esse sono ordinate in base a due fattori; le prime sono le misure presenti di default da ImageJ e dal Plugin Analyzer Particles che viene esteso dalla inner class del Plugin dell’elaborato. Dopo di che, seguono le misure implementate e aggiunte suddivise per tipologia di immagine e misura riguardante essa. Si possono anche selezionare tutte le misure o selezionare solo un gruppo di misure attraverso i check box con etichetta “Select all”. A seconda della tipologia di immagine, se in scala di grigi o se a colori, alcune misure non sono disponibili. Dopo aver selezionato le misure, selezionando la voce “Ok”, il Plugin avvia il calcolo delle misure sulle regioni interesse individuate. Viene visualizzata una ResultsTable, una tabella in cui vengono riportate le misure calcolate. Ogni colonna corrisponde ad una misura; nella prima riga differenziata da un grigio più scuro, vengono riportati i nomi delle misure. La prima colonna invece, indica il numero della regione di interesse analizzata.