Tesi

# bozza della tesi

## plugin in imagej

I componenti principali di ImageJ, che gli fanno anche da base, sono i Plugin che estendono e arricchiscono il Tool in modo semplice e efficace. Ogni elemento, anche l’interfaccia con cui accoglie l’utente, è creata attraverso i Plugin. Un Plugin si può sviluppare in Macro e Java ma in questo elaborato ci concentreremo sulle tipologie scritte con il secondo linguaggio di programmazione nominato. Qualsiasi utente può iniziare la stesura di un Plugin personalizzato. Attraverso il menu principale, scegliendo la voce “Plugins”, può iniziare la scrittura attraverso il comando new e scegliendo ilPlugin 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. In termini tecnici esse sono interfacce. PlugIn riporta un solo metodo chiamato run che prende come argomento un oggetto di tipologia String. Il metodo viene chiamato quando viene caricato il Plugin che si serve di questa interfaccia. L’argomento di classe String può essere anche vuoto, ovvero “”. Plugin Filter è leggermente più corposo. Al suo interno vi è il metodo setup. Quest’ultimo viene chiamato quando il Plugin viene caricato. Presenta due argomenti passati uno ti tipologia String che può essere vuoto e un oggetto di tipo ImagePlus che indica l’immagine attualmente attiva. Restituisce un valore di tipo int che indica se il metodo è andato a buon fine. Presenta anche un metodo run (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 all’obbiettivo proposto, o comunque non le raggruppa nello stesso Plugin. L’idea sarebbe poter sfruttare al meglio sui Plugin base 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 menù “Analyze” con la voce “Analyze Particles…”. L’elemento più importante di ParticlesAnalyzer è la possibilità di individuare più oggetti in una immagine e calcolare le features su ognuno di essi. Questo avviene attraverso un Threshould manuale, 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 menù. Particles Analyzer implementa PlugInFilter e Measurements. Measuments è una interfaccia che lo collega ad un menù in cui si possono settare alcune misure ma ai fini dell’elaborato non sarà rilevante. Date le proprietà di questo Plugin la decisione presa per implementare e raggruppare le misure già esistenti è quella di creare un nuovo Plugin che estendesse Particles Analyzer e basarsi su di esso.

### Sviluppo e storia del plugin

Il Plugin prodotto è stato aggiornato volta per volta basandosi su tre stadi; il primo stadio prevedeva la implementazione di misure in B&W, sia già calcolate sia da calcolare; il secondo stadio prevedeva la implementazione delle misure in grey; il terzo stadio e ultimo prevedeva la implementazione delle misure in RGB. Durante la stesura del codice e laricerca di metodologie da applicare per inserire 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 o solo alcune. Inoltre struttura del Plugin stesso è stato modificato più volte a causa, o grazie, a metodologie scoperte e applicate in corso d’opera che rendevano il codice comprensibile e logico. La particolarità dei Plugin di ImageJ è che si devono appoggiare completamente sulle classi o plugin già implementati su ImageJ stesso, una vera e propria libreria a parte volta solo ed esclusivamente all’utilizzo all’interno del tool.

### Struttura del Plugin

“Catch\_Parasite3” (??) è il nome attuale del Plugin elaborato. Esso implementa l’interfaccia PlugIn e al suo interno vi è un solo metodo chiamato run, e un vettore di ImagePlus di nome impVector. All’interno del metodo run viene avviato il plugin e se l’immagine è a colori viene trattata per elaborarla con i canali RGB e le nuove tre immagini derivanti dalla elaborazione vengono inserite all’interno del vettore impVector che verrà utilizzato per estrarle e calcolare le misure in RGB. Viene creato un oggetto appartenente inner class Parasite (??) che estende ParticlesAnalyzer ricavandone le proprietà necessarie per poter lavorare su più oggetti nell’immagine e altre funzioni per la visualizzazione di alcune misure implementate da ImageJ. Dopo aver creato l’oggetto Parasite, si avvia il suo setup. Se questo va a buon fine, ovvero il metodo setup rende una variabile intera uguale a “PlugInFiler.Done”, viene avviato anche il metodo run della stessa classe Parasite, dangogli come argomento l’ImageProcessor reso dall’oggetto ImagePlus attarverso il metodo getProcessor(). ImageProcessor è una classe astratta volta, come suggerisce il nome, a “processare” e trattare i dati dei pixel di un’immagine 2D. Infine viene richiamata dalla super classe Analyzer, il metodo volto a visualizzare i risultati ottenuti attraverso una ResultTable con titolo “Results”.

#### Parasite

Parasite è la inner class del Plugin principale, ovvero di Catch\_Parasite3. Viene creato un oggetto Parasite all’interno del metodo run di Catch\_Parasite3. La sua implementazione era necessaria perché estendendo la classe ParticlesAnalyzer non era banale far avviare il Plugin senza che avesse un supporto da parte di Catch\_Parasite3. Questo ultimo fa da tramite tra l’immagine e il plugin, inner classe. Parasite si occupa di estendere ParticlesAnalyzer e di implementare le nuove misure. Al suo interno vengono dichiarate una serie di variabili booleane necessarie per selezionare solo le misure scelte dall’utente e in particolare un intero “measure” che accoglierà i valori interi corrispondenti alle misure già implementate da ImageJ. Ogni valore intero ha un corrispondente specifico come misura. Seguono i metodi necessari per aggiungere le misure implementate nel plugin; molti di essi sono Override dei metodi della super classe. Il primo metodo importante è showDialog che non prende argomenti in ingresso e restituisce un valore di tipo intero. Esso è un Override e richiama il metodo della super.classe. Questo visualizzerà un menu iniziale già implementato da Particles Analyzer. Questo metodo si occupa anche di verificare se il setup è andato a buon fine. Verificando nella struttura della super classe estesa si può denotare che showDialog è chiamato all’interno del setup stesso. L’Override era necessario per aggiungere un nuovo menù per settare le misure da visualizzare. Inoltre verifica se l’immagine è a colori o meno. Questo passo è importante per visualizzare le misure per RGB oppure per non visualizzarle nel menu per la scelta delle features. All’interno del metodo viene richiamato genericDialog (??) che comporta la visualizzazione di un nuovo menu attraverso una nuova GenericDialog. Vengono visualizzate le scelte attraverso checkBox e raggruppate a seconda della implementazione o della tipologia di features da presentare. In ordine abbiamo: le features implementate da ImagJ, le features implementate durate lo studio e la creazione dell’elaborato riguardanti le misure per immagini in bianco e nero, le features riguardanti le immagini in grigio, le features riguardanti le immagini a colori. Queste ultime non appariranno se l’immagine non è a colori. Allo stesso modo le misure “min and max” per le immagini in grey non apparirà se l’immagine è a colori poiché darebbe valore zero. I valori selezionati vengono immessi nel vettori booleani di appoggio riguardanti le misure di ImageJ, le misue in Bianco e Nero, misure in grigio e misure in RGB. Il metodo setMeasureExtended() si occuperà di recuperare i valori booleani dei vettori; per le misure di ImageJ dipendentemente dal valore positivo o negativo verranno aggiunte le costanti alla variabile intera measure, che verrà poi passata a Analyzer.setMeasurements(measure). Per le misure implementate dall’elaborato verranno presi i valori dipendentemente dall’indice e immessi nelle variabili booleani i quali indicano attraverso il loro nome quale misura verrà calcolata a seconda del suo valore vero o falso; per esempio “doConvexArea” se verrà settato come vero nella ResultTable apparirà tale valore. L’Override più importante è del metodo saveResults; la super classe si occupa di creare l’oggetto Roi che prenderà il nome roi nel metodo, l’area di interesse che occupa l’oggetto singolo analizzato nella immagine e il suo ImageStatistics ovvero un oggetto che si occupa del calcolo delle misure di natura statistica dell’oggetto. Dopo aver richiamato il metodo della super classe, vengono dichiarate alcune variabili necessarie per il calcolo di alcune misure; un oggetto della classe Polygon reso dall’oggetto roi attraverso il metodo getConvexHull(); in questo caso ImageJ usa una classe di Java Polygon per trattare con il convexHull della regione di interesse. Grazie alla creazione di tale oggetto, ne viene calcolata area e perimetro del convex hull stesso, chiamato un metodo della classe Parasite, getArea e getPerimeter. Per questione di comodità viene dichiarata e settata la variabile fert, un vettore contenente i dati relativi al feret attraverso un metodo della roi e il perimetro sempre attraverso un metodo. Ultimo ma non meno importante viene anche dichiarato un vettore hist e popolato dall’istogramma della roi, attraverso la restituzione da stats, il suo ImageStatistics collegato. Con una serie di if consecutivi, viene verificata la necessità di calcolare le features attraverso i booleani settati precedentemente. Ognuna di esse ha una formula a sé spesso applicata nella stampa. Solo le features Haralick Ratio, Bending Energy, Entropy, IntensitySum, Uniformity, Smothness si è preferito farle a parte in un metodo per avere un codice più pulito e comprensibile. Se l’immagine è in RGB si controllano anche le misure relative a quella tipologia. Dunque si crea un oggetto ImageStatistics dalle immagini ricavate inizialmente nel plugin, 0 red 1 green e 2 blue, e si setta la roi ad esse. Per il resto, la verifica se una features è da implementare o meno è uguale alle precedenti. Dopo l’Override di saveResults seguono i metodi di appoggio per calcolare l’area e il perimetro del convexHull, Bending Energy, HeralickRatio, Entropy, InsentisitySum, Uniformity, Smothness. A seguire una serie di metodi di appoggio per la distanza, calcolo del logaritmo in base due e metodologie dei vettori.

### come usare il plugin

Nel seguente menù, l’utente può decidere la grandezza minima dell’oggetto analizzato e a seguire quella massima e la sua circolarità. Attraverso il menu a tendina si può scegliere il trattamento della immagine finale. Le scelte 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, count mask (??), 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 che si possono settare attraverso checkbox sono display results per visualizzare appunti i risultati, clear results per cancellare i risultati precedenti e summarize, Add to Manager, Exclude on Edges, Include holes, records starts, in situ Show.