Rilevazione automatica di microtubuli astrali in immagini di microscopia a fluorescenza

UNIVERSITÀ DEGLI STUDI DI TORINO

Dipartimento di Scienze Mediche
Corso di Laurea in Medicina e Chirurgia

Tesi di Laurea

Rilevazione automatica di microtubuli astrali in immagini di microscopia a fluorescenza

Relatore:
prof. Ferdinando Di Cunto

Candidata:
Gabriella Olmo

Settembre 2016

Ringraziamenti

Si ringrazia la Dott. Marta Gai per aver fornito i dati di test e di controllo, e per le utilissime discussioni sull'argomento.

Sommario

Questa tesi si colloca nel contesto dell'elaborazione automatica di immagini di microscopia a fluorescenza, e si propone di studiare algoritmi utili per identificare e seguire automaticamente l'evoluzione dei microtubuli astrali. L'uso di metodiche di elaborazione e analisi di immagini sta assumendo un ruolo sempre più importante nelle scienze biomediche, in quanto è possibile generare enormi masse di dati che contengono informazioni dinamiche su strutture subcellulari *in vivo*. Lo studio dell'evoluzione dinamica di tali strutture (di cui i microtubuli rappresentano un significativo esempio) permette di ottenere informazioni di fondamentale importanza nei campi della biologia molecolare e della medicina. Per esempio, è noto come il corretto orientamento del fuso mitotico sia un fattore importante che regola la differenziazione cellulare durante l'embriogenesi, e che quindi mutazioni in geni che interessano tale processo (ASPM - *abnormal spindle-like microcephaly associated*, CIT - *citron kinase*) siano responsabili di casi di microcefalia [1]. Entrambi questi geni sono coinvolti nell'organizzazione dei microtubuli astrali, e interagiscono tra loro in maniera non completamente compresa. Il presente lavoro si colloca quindi nell'ambito di un'attività di ricerca volta a chiarire aspetti legati ad anormale nucleazione e instabilità dei microtubuli astrali in cellule in cui questi geni siano stati soppressi tramite specifici siRNA. Tale attività è sponsorizzata dall'associazione Telethon (grant n. 12095) e dall'Associazione Italiana per la Ricerca sul Cancro (AIRC - grant n. IG 17527), il che testimonia le grandi ricadute mediche, attuali e potenziali, di questo tipo di ricerca.

L'analisi manuale volta a rilevare la presenza in immagini di determinate strutture e la loro evoluzione in fotogrammi successivi rappresenta un lavoro lungo e pesante, caratterizzato da un basso livello di riproducibilità. Di fatto, esso costituisce un collo di bottiglia, che limita nella pratica la possibilità di analizzare i tantissimi dati che si possono produrre ad ogni esperimento. Di conseguenza, la necessità di algoritmi che permettano di automatizzare tale lavoro si fa sempre più impellente. D'altro canto, se i problemi di *target detection* e *pattern recognition* sono ben noti e studiati da tempo in campi quali l'analisi radar o sonar, l'aeronautica, il telerilevamento, gli algoritmi sviluppati per tali applicazioni sono difficili da applicare nel settore della microscopia ottica a fluorescenza. Infatti, in questo caso

ci si trova di fronte a immagini affette da elevati tassi di rumore, in cui si devono identificare "oggetti" di dimensioni inferiori al limite di risoluzione dello strumento, e quindi percepibili come "macchie" luminose, senza che sia possibile fare affidamento su informazioni morfologiche. Inoltre, nel caso in cui sia necessario seguire l'evoluzione nel tempo di macromolecole (come i microtubuli), queste spesso sono caratterizzate da modelli di movimento non lineari, inclusa la possibilità che l'oggetto da identificare sparisca in una determinata immagine per poi eventualmente ricomparire successivamente, e ciò rende complesso seguirne l'evoluzione nel tempo.

L'organizzazione di questo lavoro di tesi è la seguente.

- Innanzitutto, si è analizzato lo stato dell'arte relativo al problema della identificazione e dell'inseguimento di oggetti in immagini di microscopia a fluorescenza, con particolare attenzione agli algoritmi aventi come oggetto l'analisi di microtubuli.

- Sono stati evidenziati i punti critici del rilevamento di oggetti di interesse in ciascuna singola immagine, e del successivo inseguimento dei medesimi nelle immagini acquisite a istanti di tempo successivi. Sono stati sottolineati gli aspetti che rendono il problema differente rispetto ad applicazioni simili in altri settori: il fatto di tracciare oggetti di dimensioni al di sotto del limite di risoluzione dello strumento in immagini affette da basso rapporto segnale-rumore, il che rende spesso gli oggetti significativi quasi indistinguibili dallo sfondo; il fatto che il rumore sia diverso dal classico rumore additivo gaussiano bianco, ma modellabile piuttosto come poissoniano, legato all'intrinseco processo di rilevamento di una sorgente discreta con basso numero di fotoni emesso. Inoltre, le particelle presentano un movimento complesso, e possono apparire, scomparire, apparire nuovamente in immagini successive. Gli oggetti da inseguire possono essere estremamente numerosi, il che può portare a traiettorie che si incrociano o si fondono e poi successivamente separano, situazione normalmente non gestita nelle comuni applicazioni.

- Per ciascuno dei principali problemi elencati, sono state analizzate le soluzioni proposte in letteratura, evidenziandone aspetti positivi e aspetti critici in termini di prestazioni e robustezza. Sono state selezionate le seguenti tecniche:

  Per il rilevamento: *Laplacian of Gaussian, Difference of Gaussians*, analisi multiscala (trasformata *wavelet*), trasformata di Anscombe per la regolarizzazione della varianza del rumore impulsivo seguita da filtraggio di Wiener.

  Per l'inseguimento: filtro di Kalman e sue varianti.

- È stata svolta un'analisi statistica dei dati a disposizione, relativi a esperimenti condotti con microscopio confocale, per valutare il rapporto segnale-rumore ed

evidenziare la natura del rumore presente nelle immagini. Sulla base di questa analisi, sono stati identificati gli algoritmi di rilevamento e inseguimento che più sembrano adatti al caso specifico di studio.

- È stata svolta una ricerca di strumenti software disponibili per poter sperimentare gli algoritmi selezionati. A tale fine, è stato identificato un *toolbox* MATLAB denominato U-track, sviluppato e reso disponibile dalla *Harvard Medical School*, che include alcune opzioni di rilevamento e di inseguimento, e numerose funzioni ausiliarie per la visualizzazione e il calcolo di statistiche relative ai risultati ottenuti. Tale *toolbox* è stato installato, sono stati risolti alcuni problemi tecnici e lo si è portato a un livello di funzionamento stabile.

- Sono state svolte campagne sperimentali di analisi dei dati reali a disposizione, usando diversi algoritmi di rilevamento e diversi parametri (dai quali si è riscontrata una forte dipendenza). Vale la pena di notare come la maggior parte degli algoritmi proposti in letteratura siano stati validati principalmente su dati sintetici, o su sottoinsiemi limitati di dati reali. Di conseguenza, le prestazioni riportate in letteratura frequentemente non sono ritrovate quando si lavora con dati relativi a esperimenti complessi.

- Sono state apportate modifiche agli algoritmi esistenti, specialmente relative alla parte di rilevamento, che si è rivelata la più critica. In particolare, si sono pre-elaborate le immagini mediante la trasformata di Anscombe seguita da filtro di Wiener, per limitare l'impatto del rumore di Poisson. Si è aggiunto uno stadio di deconvoluzione specifico per rumore poissoniano, facente uso dell'algoritmo di Lucy-Richardson, e si è eseguito il confronto con il più generale metodo di deconvoluzione *blind*.

- I risultati ottenuti sono stati valutati sia qualitativamente che numericamente, mediante confronto con i numeri medi di "comete" e di tracce rilevate manualmente dal personale del Laboratorio, e delle relative informazioni statistiche (velocità, durata, lunghezza media delle tracce).

- Sono state tratte valutazioni conclusive, ed è stato tracciato un possibile sviluppo futuro per un'attività che naturalmente va ben al di là degli orizzonti di una tesi di laurea, sia per la complessità della tematica sia per il numero crescente di applicazioni promettenti che possono beneficiare della disponibilità di un supporto software.

Indice

Capitolo 1

Introduzione

Il rilevamento di oggetti di interesse in immagini digitali (*detection*), e l'analisi del loro movimento in sequenze temporali di immagini prese a istanti di tempo successivi (*tracking*), è sempre stato un argomento importante di ricerca, con enormi ricadute in settori quali la videosorveglianza, il posizionamento globale (GPS), la guida assistita di veicoli, il telerilevamento per applicazioni sia militari che civili, il controllo della deforestazione o delle coltivazioni da immagini satellitari ottiche o radar. Sono inoltre importanti le applicazioni multimediali, l'analisi di immagini mediche e di biologia molecolare. Quest'ultimo settore di applicazione, a cui fa riferimento il presente lavoro di tesi, è da una parte estremamente importante e ricco di potenziali ricadute sia scientifiche che sociali ed economiche in termini di miglioramento della salute umana, dall'altra molto complesso, con caratteristiche peculiari che rendono molto problematica l'estensione a questo tipo di applicazione delle tecniche sviluppate in settori differenti.

Attualmente, lo strumento di gran lunga più importante per studiare sistemi sub cellulari dinamici *in vivo* è ancora la microscopia ottica [2]. Le moderne tecniche di microscopia consentono la visualizzazione di tessuti, cellule e strutture macromolecolari *in vivo*, attraverso la fusione di proteine specifiche con marcatori fluorescenti come la GFP (*green fluorescent protein*), e rendendo possibile l'analisi dinamica di queste strutture. Esprimendo una specifica proteina di interesse fusa con una FP (*fluorescent protein*), è possibile associare una fluorescenza nel campo del visibile a uno specifico gene, consentendo quindi di visualizzare otticamente virtualmente qualsiasi proteina di interesse in una cellula vitale. Marcando proteine diverse con varianti diverse di FP, attive a diversi spettri di assorbimento ed emissione, è possibile effettuare il rilevamento simultaneo di molte proteine, rimarcando così la loro eventuale interazione. Infatti, i sistemi biomolecolari sono essenzialmente sistemi dinamici, e la ricerca è sempre maggiormente orientata a rivelare le relazioni sia spaziali che temporali che sussistono tra proteine e complessi macromolecolari, e fornire metodi per manipolarle [2]. Questi studi generano masse ingenti di dati, la

cui elaborazione sta diventando il collo di bottiglia dell'intero processo. Di conseguenza, la capacità di gestire questi dati e sfruttarne appieno le potenzialità per descrivere i processi biologici sottostanti a livello quantitativo oltre che qualitativo, costruendo modelli accurati delle strutture dinamiche, sta diventando essenziale e di grande significato biologico. Si noti come attualmente buona parte dell'analisi delle immagini generate dalla microscopia ottica a fluorescenza venga effettuata manualmente. Si tratta di un lavoro gravoso e tedioso, che sottrae energie ad attività di maggiore rilevanza, e che per giunta non è generalmente ben riproducibile, essendo affetto da una polarizzazione dovuta all'esperienza dell'operatore. Questo è uno dei motivi che rendono difficoltoso il confronto dei risultati di esperimenti condotti da diversi gruppi di ricerca, unitamente alle numerose diversità di condizioni sperimentali, molto difficili da uniformare dato l'elevato numero di variabili in gioco.

1.1 Considerazioni generali

Come accennato in precedenza, l'elaborazione di immagini di biologia molecolare pone una serie di difficoltà specifiche.

La prima difficoltà è l'intrinseca, limitata risoluzione spaziale del microscopio ottico. Infatti, nonostante i notevoli progressi tecnologici, la risoluzione spaziale anche dei migliori microscopi odierni è piuttosto scadente, e si aggira intorno ai 100 nm [3], assai maggiore delle dimensioni tipiche delle strutture sub-cellulari di interesse (decine di nm). Di conseguenza, tali strutture presenteranno un aspetto "limitato dalla diffrazione", ovvero si presenteranno come macchie indistinte, essendo le loro dimensioni al di sotto del limite di risoluzione spaziale del microscopio. Come conseguenza, un algoritmo di riconoscimento automatico non potrà sfruttare le informazioni morfologiche, ossia legate alla forma dell'oggetto da identificare, per distinguere il medesimo da strutture irrilevanti che costituiscono lo sfondo, oppure da locali picchi di rumore. Se si applicano algoritmi tipici di rilevamento automatico all'analisi di immagini di microscopia a fluorescenza, in assenza di provvedimenti correttivi ci si deve aspettare di conseguire un elevato tasso di falsi positivi (FPR, *false positive rate*), dovuti al frantendimento di oggetti irrilevanti o picchi di rumore che vengono interpretati come oggetti di interesse, oppure, se l'algoritmo viene tarato in modo da limitare FPR, da un'elevata probabilità di mancato rilevamento, dovuta al fatto che l'algoritmo risulta polarizzato verso il riconoscimento degli oggetti più chiaramente distinguibili dallo sfondo [3]. La risoluzione di un microscopio è misurata mediante la sua risposta all'impulso, o *point spread function* (PSF), che assume il significato della forma d'onda bidimensionale o tridimensionale, a seconda della dimensionalità del problema[1], che viene rappresentata nell'immagine quando

[1] In questa tesi, per semplicità di notazione e ove non diversamente specificato, ci si riferirà al caso di immagini bidimensionali.

il microscopio illumina un oggetto puntiforme. Nel caso di un microscopio confocale con apertura circolare, la PSF si può rappresentare come [2]

$$PSF(r,z) = \left| \int_0^1 2J_0(\alpha r\rho) exp(-2j\gamma z\rho^2)\rho d\rho \right|^2$$

dove

$$\alpha = \frac{2\pi NA}{\lambda}$$

e

$$\gamma = \frac{\pi NA^2}{2\lambda}$$

con $r = \sqrt{(x^2 + y^2)}$ che rappresenta la distanza radiale dall'asse ottico, z la distanza assiale dal piano focale, J_0 la funzione di Bessel del primo tipo di ordine zero, NA l'apertura numerica della lente, e λ la lunghezza d'onda della luce emessa dall'oggetto sotto esame. Questa funzione, nella maggior parte delle applicazioni pratiche, si può approssimare con una forma d'onda gaussiana bidimensionale [3]. Questo fatto ha ripercussioni dirette sugli algoritmi di rilevamento oggetti, come sarà discusso nei prossimi capitoli.

Un secondo problema è dovuto al fatto che la qualità delle immagini è generalmente scadente, con un rapporto segnale-rumore (SNR, *signal-to-noise ratio*) molto basso. Questo problema è particolarmente acuto nelle applicazioni che coinvolgono acquisizione di immagini *in vivo*, situazioni in cui l'intensità dell'illuminazione è ridotta al minimo per limitare il danneggiamento della cellula e il fenomeno del *photobleaching*, ovvero la progressiva perdita di fluorescenza a seguito della stimolazione dei fluorofori da parte della sorgente laser di eccitazione, che ne comporta modificazioni covalenti. Per lo stesso motivo, ovvero per evitare i suddetti fenomeni di danneggiamento e *photobleaching*, la frequenza di acquisizione delle immagini o *frame rate* è mantenuta bassa, specie se confrontata con la dinamica tipica degli oggetti di dimensioni nanometriche. Infatti, da una parte è necessario acquisire immagini per un intervallo di tempo sufficientemente lungo da consentire l'inseguimento di oggetti così piccoli, dall'altra è necessario preservare l'integrità della cellula e limitare il *photobleaching*, e di conseguenza sarà necessario rallentare il *frame rate* [4]. Questo fatto è in contrasto con la necessità di affidarsi principalmente alla conoscenza del moto dell'oggetto per poterlo inseguire, dal momento che non è possibile sfruttare il suo profilo spaziale. Gli oggetti nanometrici spesso presentano dinamiche di movimento molto complesse ed erratiche, il che implica il fatto che oggetti possano eventualmente scomparire (per esempio, perché il loro movimento li ha portati fuori dal piano focale del microscopio), ricomparire, invertire la direzione di movimento, incrociare le proprie traiettorie ecc. Inoltre, il numero di oggetti presenti nella singola immagine può essere assai elevato, rendendo il problema ancora maggiormente complesso.

Per quanto concerne la statistica del processo di rumore, mentre nella maggior parte delle applicazioni esso si può considerare come rumore additivo gaussiano bianco (AWGN, *additive white gaussian noise*) dovuto all'agitazione termica degli elettroni di conduzione dell'elettronica di rilevamento, nel caso in questione invece si deve parlare di *rumore fotonico intrinseco*, dovuto alla natura casuale del processo di emissione dei fotoni, e indipendente dal rilevatore. Data la sua natura, intrinsecamente legata al processo di emissione dei fotoni, il rumore fotonico (anche detto *rumore granulare, rumore shot, rumore di Poisson* in quanto segue una distribuzione poissoniana) è dipendente dal segnale.

Il rumore *shot* è quindi dovuto alla natura particellare della luce, per cui qualunque dispositivo atto a contare il numero di fotoni in arrivo (fotomoltiplicatore, fotodiodo a valanga) sarà affetto da questo tipo di rumore. In condizioni normali, il numero di fotoni emesso da un laser per generare una macchia luminosa è così grande che l'effetto di quantizzazione è trascurabile, ovvero il numero di fotoni che genera una macchia di una data intensità non è costante, ma comunque fluttua in maniera trascurabile tra un esperimento e l'altro. Ciò non è più vero se il laser viene mantenuto a livelli di emissione molto bassi, come nelle applicazioni di microscopia a fluorescenza per le ragioni precedentemente descritte. In questo caso, il rumore *shot* può essere dominante. Va da sé che questo rumore possa essere diminuito solamente aumentando l'intensità luminosa. Infatti, l'ampiezza del rumore *shot* cresce come la radice quadrata del numero di fotoni emessi per unità di tempo, $\sqrt{N}$. Tuttavia, siccome l'intensità del segnale utile cresce come N, ne deriva che il rapporto segnale-rumore aumenta all'aumentare di N:

$$SNR = \frac{N}{\sqrt{N}} = \sqrt{N}$$

Tuttavia, come già discusso, aumentare N non è possibile se non a spese di un'aumentata fototossicità o di un maggiore effetto di *photobleaching*.

Per valori grandi di N, la distribuzione di Poisson tende a una distribuzione gaussiana; i singoli fotoni non sono più osservati individualmente, e il rumore *shot* diventa indistinguibile dal rumore gaussiano. Infatti, se N è molto grande, anche SNR è molto grande, e altre sorgenti di rumore (come il rumore termico) dominano rispetto alle fluttuazioni relative di N. Bisogna notare come l'elettronica di rilevamento a sua volta aggiungerà rumore AWGN, dando come risultato il rumore misto gaussiano e di Poisson, che costituisce il modello più realistico di rumore che colpisce le immagini di microscopia ottica [5]. Qualora N sia grande, il processo di rumore dominante sarà il rumore gaussiano bianco, e quindi si potranno usare le tecniche di miglioramento di SNR comunemente adottate in questa situazione.

Infine, un terzo fattore che complica l'analisi automatica di queste immagini è la loro spiccata variabilità, dovuta da una parte alla intrinseca eterogeneità dei sistemi biologici, dall'altra a una mancanza di standardizzazione dei processi di

acquisizione. Questo comporta che immagini relative allo stesso fenomeno fisico di interesse possano essere caratterizzate da livelli di qualità molto differenti [2].

1.2 Identificazione e inseguimento dei microtubuli

I microtubuli (MT) sono polimeri del citoscheletro rigidi, labili e polari, caratterizzati da un comportamento dinamico molto caratteristico. Essi infatti alternano fasi di polimerizzazione (crescita), di depolimerizzazione (*shrinkage*), di pausa, e molte delle funzioni cellulari associate ai MT dipendono dallo scambio dinamico tra una fase e l'altra.

Le unità elementari che costituiscono i MT sono eterodimeri di due polipeptidi globulari di dimensioni 4 x 5 x 8 nm e peso molecolare 55 kDa. Ciascun dimero è costituito da una subunità di α-tubulina e una di β-tubulina, entrambe facenti parte della famiglia delle tubuline [6]. I dimeri polimerizzano in tredici protofilamenti, affiancati e sfasati a spirale, formanti un tubulo cavo con diametri esterno e interno di 25 nm e 15 nm rispettivamente. La polimerizzazione segue uno schema polare, con la subunità α orientata verso la cosiddetta *estremità minus* (-), e la subunità β verso l'*estremità plus* (+). Le due estremità presentano quindi diversità strutturali e chimiche, nonché di funzione. I MT polimerizzano e depolimerizzano in continuazione all'interno della cellula, dando origine a un fenomeno di rinnovamento continuo chiamato *treadmilling*. Questo è causato dall'affinità dei MT per la tubulina libera nel citoplasma. La tubulina polimerizzata è legata a un ribonucleotide guaninico. Man mano che avanzano verso l'estremità *minus*, le molecole di tubulina idrolizzano la molecola di GTP, a maggior affinità, in GDP a minore affinità. L'equilibrio tra le due estremità determina la velocità di crescita o decrescita. Sebbene l'evento sia sostanzialmente autonomo, diversi fattori posso influenzare la polimerizzazione o depolimerizzazione dei MT, tra cui la concentrazione di Ca2+ e la temperatura. Inoltre, alcuni composti favoriscono uno sfaldamento dei microtubuli, mentre altri sono capaci di stabilizzarli. Una terza proteina, la γ-tubulina, interagisce con la β-tubulina in un processo detto *nucleazione*, inducendo la polimerizzazione del MT. Tale processo di nucleazione avviene nei cosiddetti *microtubules organization center* (MTOC), che tipicamente coincidono con i centrosomi.

I MT svolgono moltissime funzioni essenziali per la cellula, tra cui l'organizzazione e trasporto intracellulare e funzioni di stabilità meccanica. Essi costituiscono l'impalcatura interna di flagelli e ciglia, e regolano il transito di organuli e vescicole all'interno della cellula. Essi inoltre svolgono un ruolo importante nella secrezione degli ormoni della tiroide e del pancreas.

Durante la divisione cellulare, i MT presiedono alla costituzione del fuso mitotico. Durante la profase, dai due centrosomi si dipartono MT che si dispongono a raggiera, con le estremità *plus* rivolte verso la periferia e le estremità *minus* in vicinanza del centrosoma (*microtubuli astrali*). Quando le due coppie di centrioli si allontano tra di loro durante la fase M in profase, in mezzo a loro si formano fasci di microtubuli che costituiscono il fuso mitotico; alcuni di questi si legheranno ai cromosomi (*microtubuli cinetocorici*), mentre gli altri, detti *microtubuli polari*, hanno la funzione di allontanare i due poli del fuso mitotico durante l'anafase B, una volta che tutti i cromosomi sono stati legati dai MT cinetocorici.

Le diversità funzionali dei MT sono dovute a diverse caratteristiche, fra le quali la presenza di numerose proteine associate ai microtubuli (MAP, *microtubule associated protein*), che convertono la rete instabile di MT in una ossatura relativamente permanente con proteine che incapsulano l'estremità crescente della tubulina impedendone la depolimerizzazione. Alcune di queste differenze predominano in certe cellule tumorali ed alcune sono associate allo sviluppo della resistenza al farmaco. Vi sono pure diverse proteine motrici che regolano il trasporto di vescicole e organelli lungo i MT: la dineina, che permette il moto lungo i MT verso il centro della cellula, e la chinesina, che promuove il movimento ameboide lungo il MT verso la periferia cellulare. Alcuni farmaci agiscono alterando la dinamica dei MT; essi comprendono i farmaci antineoplastici della classe del taxolo, quali ad esempio il paclitaxel. La colchicina induce una completa depolimerizzazione dei MT, e ciò provoca un potente effetto antimitotico nella cellula in quanto non viene generato il fuso mitotico. Questo farmaco, eccessivamente tossico per consentirne un uso antineoplastico, viene utilizzato per il trattamento della gotta e delle pericarditi per il suo potente effetto anti-infiammatorio e per bloccare l'attività di fagocitosi. Il mebendazolo danneggia e impedisce lo sviluppo dei microtubuli in alcune famiglie di vermi; per questo appartiene ai farmaci antielmintici, ed è stato sperimentato anche nella lotta contro il carcinoma del polmone e il carcinoma adrenocorticale [7].

Il corretto orientamento del fuso mitotico durante l'embriogenesi è anch'esso ottenuto grazie a una corretta organizzazione dei MT astrali. Questi, a loro volta, subiscono processi di nucleazione e stabilizzazione regolati da geni quali ASPM (*abnormal spindle-like microcephaly associated*, MCPH5) [1], il gene maggiormente coinvolto nella genesi della microcefalia umana primaria autosomica recessiva (*human primary microcephaly*, MCPH), che interagisce in modo non completamente chiaro con la *citron kinase (CITK)*, un'altra proteina filogeneticamente conservata responsabile di microcefalia severa. La corteccia cerebrale embrionale è inizialmente composta da un singolo strato di cellule neuroepiteliali pluripotenti, che si espandono tramite un processo di divisione simmetrica. Successivamente, esse danno origine a cellule della glia radiale, che a loro volta possono continuare a dividersi in modo simmetrico, oppure possono dividersi in modo asimmetrico, dando origine a una cellula

figlia destinata alla differenziazione terminale. È noto come queste cellule pluripotenti si dividano secondo un piano di clivaggio perpendicolare alla superficie del neuro-epitelio, mentre l'evoluzione verso la differenziazione terminale sia accompagnata da una deviazione di questo asse di clivaggio. Chiaramente, un troppo precoce instaurarsi della differenziazione cellulare comporta una scarsa massa cerebrale definitiva, e quindi una microcefalia, patologia che si manifesta con uno scarso diametro del cranio, accompagnato da ritardo mentale lieve o moderato. Il posizionamento del fuso mitotico dipende dalla formazione di legami molecolari tra la corteccia cellulare, ricca di actina, e i MT astrali, la cui nucleazione è per l'appunto regolata dai summenzionati geni ASPM e CITK (oltre che da molte altre proteine). Non è chiaro se la relazione tra scorretto orientamento del fuso mitotico e differenziazione terminale sia di tipo causale, o semplicemente correlativa. Tuttavia, il coinvolgimento di questi geni nella microcefalia è appurato. Il lavoro nell'ambito del quale si colloca questa tesi è volto a chiarire aspetti legati ad anormale nucleazione e instabilità dei microtubuli astrali in cellule in cui questi geni siano stati soppressi tramite specifici siRNA. Tale attività è sponsorizzata dall'associazione Telethon (grant n. 12095) e dall'Associazione Italiana per la Ricerca sul Cancro (AIRC - grant n. IG 17527). I materiali e i metodi usati per ottenere e trattare le colture cellulari sono descritti in [1]. Si tratta comunque di colture di cellule HeLa che esprimono proteine CITK e ASPM marcate GFP, sottoposte a una transfezione con sequenze siRNA adeguate. Inoltre, neuroblasti di Drosophila con allele dck mutante [1] opportunamente preparate, sono usate per studiare l'orientamento del fuso in mitosi. Per quanto riguarda le immagini, si è usato un microscopio confocale Leica TCS SP5-AOBS a 5 canali (Leica Microsystems) attrezzato con un diodo fotorilevatore a 205 nm e un laser DPSS a 561 nm. Le sequenze sono state acquisite ad intervalli di cinque minuti usando un obiettivo con immersione in olio e una risoluzione di $0.108 \cdot 0.108$ μm.

Per quanto riguarda la misura del comportamento dinamico dei MT in immagini di microscopia a fluorescenza, è possibile operare mediante tubulina marcata da fluorofori e iniettata o espressa in cellule viventi [8]. Tuttavia, stante la densità elevatissima di MT nel corpo cellulare, in questo modo risulta possibile soltanto analizzare il comportamento dei MT alla periferia delle cellule. Più recentemente, si sono sviluppate tecniche che permettono la visualizzazione dell'estremità in crescita dei MT, grazie all'espressione di proteine quali EB1 o EB3 che legano l'estremità *plus* del polimero, fuse con proteine fluorescenti [9] (Fig. 1.1). In questo modo, le fasi di crescita del MT sono evidenziate come "comete" che si generano e propagano lungo la traiettoria di crescita del MT. Le proteine +TIP sono un sottoinsieme delle MAP, e si legano direttamente alle estremità *plus* del MT in fase di crescita. Di conseguenza, uno svantaggio di analizzare i MT usando immagini marcate +TIP è che è possibile rilevare direttamente solo le fasi di crescita, e non quelle di pausa o di *shrinkage*, che possono solo essere inferite dall'analisi delle traiettorie evidenziate durante le fasi di crescita. Ad esempio, se due traiettorie risultano collineari e

separate da un breve intervallo di tempo, è assai probabile che appartengano allo stesso MT che ha subito una pausa. Di conseguenza, le traiettorie complete del MT possono essere ricostruite da immagini di comete relative alla sola fase di crescita mediante opportuni algoritmi software, come sarà discusso nei prossimi capitoli.

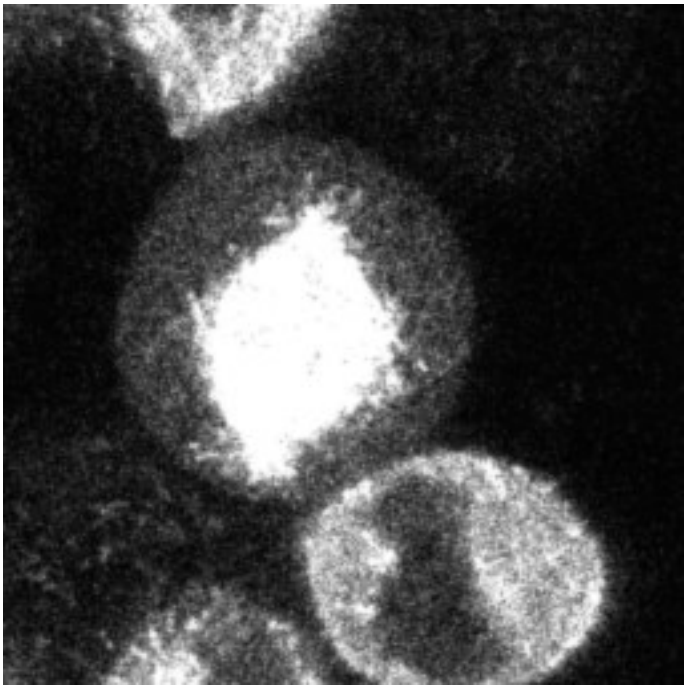


Figura 1.1. Immagine da microscopio confocale rappresentante cellula in mitosi, con MT marcati con proteina fluorescente EB3

Capitolo 2

Identificazione e inseguimento di microtubuli: stato dell'arte

In questo capitolo si descriveranno gli algoritmi proposti in letteratura per rivelare la presenza di microtubuli in immagini di microscopia a fluorescenza e inseguirne l'evoluzione temporale, identificandone la traiettoria e comprendendo le fasi di crescita, di *shrinkage* e di pausa. Si ipotizzerà, ove non diversamente specificato, che le immagini siano ottenute marcando l'estremità *plus* del MT con proteine fluorescenti +TIP, come EB1-GFP o EB3-dtTomato [9]. Di conseguenza, un software di analisi automatica comporta innanzitutto l'identificazione delle "comete" in ciascuna singola immagine della pila temporale di immagini che costituiscono l'esperimento (elaborazione spaziale), e successivamente l'inseguimento delle comete rilevate attraverso immagini successive (elaborazione temporale). In tutti gli articoli pubblicati fino ad ora sull'argomento, questi due stadi, che chiameremo rispettivamente *rilevamento* e *inseguimento*, sono mantenuti distinti, ed eventualmente ripetuti iterativamente. Di conseguenza, nel seguito riassumeremo lo stato dell'arte separatamente per questi due argomenti.

2.1 Rilevamento automatico di comete

Le estremità dei MT marcate con +TIP fluorescenti appaiono come comete, o particelle luminose di dimensioni confrontabili con o inferiori al limite di risoluzione del microscopio, che variano in dimensioni, in intensità e in morfologia sia nel tempo che in regioni differenti della cellula. Anche lo sfondo delle immagini è soggetto a fluttuazioni spaziali e temporali. A causa di tutto ciò, e considerato il basso SNR, le comete sono spesso difficilmente distinguibili dallo sfondo. Questo fa sì che l'operazione di rilevamento delle comete non si possa ricondurre a operazioni

banali quale per esempio l'applicazione di una soglia sull'intensità luminosa dei pixel (sopra soglia: oggetto rilevato; sotto soglia: rumore o oggetto spurio dovuto a variazioni casuali dello sfondo). Un approccio di questo genere darebbe certamente un tasso inaccettabile di falsi positivi, o al contrario di falsi negativi a seconda dell'impostazione del valore di soglia.

La maggior parte dei lavori presentati in letteratura scompongono concettualmente il problema in un certo numero di passi, che non necessariamente sono realizzati separatamente ma possono essere integrati nelle realizzazioni pratiche:

1. Riduzione del rumore.
2. Miglioramento del segnale.
3. Effettiva identificazione delle comete.

2.1.1 Riduzione del rumore

Come discusso nel capitolo 1, le immagini sono generate da un sistema ottico seguito da un fotorivelatore, che converte il flusso di fotoni incidenti in un flusso di elettroni, e dalla circuiteria elettronica associata a quest'ultimo. Di conseguenza, i processi di rumore che disturbano queste immagini sono almeno due: il rumore additivo gaussiano bianco (AWGN) indotto dalla circuiteria elettronica, e il processo di Poisson, proporzionale al numero dei fotoni ricevuti nell'unità di spazio e di tempo. Quest'ultima sorgente di rumore è *dipendente dal segnale*, e quindi non può essere gestita da approcci tradizionali, che normalmente assumono un modello gaussiano additivo. Infatti, anche se il primo metodo per trattare rumore moltiplicativo è stato l'uso di un filtro lineare, ovvero il filtro di Wiener che verrà descritto successivamente, si è osservato come un filtraggio lineare applicato in questo contesto induca un'eccessiva limitazione del segnale. In particolare i dettagli risultano esageratamente smussati, dato che il segnale utile risulta legato al processo di rumore in modo non lineare. Si deve quindi ricorrere a un'elaborazione equiparabile a un filtraggio non lineare.

Metodi basati sulla stabilizzazione della varianza

Un approccio è quello di trasformare il rumore moltiplicativo in rumore additivo, e successivamente sfruttare le metodiche utili per rumore gaussiano additivo (AWGN). Per esempio, la conversione logaritmica, valida per il rumore *speckle*[1] [10] oppure la

[1] Il rumore *speckle*, o granulare, degrada la qualità dei segnali laser a elevata intensità, radar, radar ad apertura sintetica, ultrasuoni o tomografici, ed è presente qualora una superficie, illuminata a una certa lunghezza d'onda λ, presenti irregolarità alla stessa scala di λ. Immagini ottenute da tali superfici mediante un sistema di illuminazione coerente, quale laser o ultrasuoni, sono affette da rumore dovuto all'interferenza dell'onda di ritorno all'apertura del trasduttore; si tratta di una

trasformata di Anscombe [11] conseguono questo scopo. Infatti, se si modella il segnale osservato (per semplicità considerato monodimensionale) come

$$x[n] = s[n] \cdot \eta[n],$$

dove $s[n]$ è il segnale utile e $\eta[n]$ è il rumore moltiplicativo, applicando un operatore logaritmico si ottiene

$$log\{x[n])\} = log\{s[n]\} + log\{\eta[n]\}.$$

Il nuovo processo di rumore, $log\{\eta[n]\}$, risulta quindi additivo. Per quanto riguarda la sua statistica, sovente si assume che essa sia gaussiana. Tuttavia, è stato dimostrato come questa ipotesi non sia realistica, in particolare nel caso in cui $\eta[n]$ sia un processo di Poisson [12]. Di conseguenza, l'applicazione di metodiche adatte per processi AWGN non è teoricamente giustificata, anche se rimane comunque un'approssimazione usata con buoni risultati nel caso di rumore *speckle*.

Un metodo più adatto al caso di rumore di Poisson è l'uso della trasformata di Anscombe, che fa parte dei cosiddetti *metodi di stabilizzazione della varianza*. Questi consistono nell'applicare una trasformazione opportuna, in modo che i dati trasformati abbiano una varianza che non dipende dal valore atteso ("stabilizzata"). Nel caso di distribuzione di Poisson, la varianza σ^2 è uguale al valore atteso μ, e quindi aumenta all'aumentare dell'intensità del segnale. Questo fa sì che l'applicazione di soglie fisse o filtri lineari per la riduzione del rumore non sia un procedimento corretto. La trasformata di Anscombe esegue la seguente operazione:

$$x \to y = 2\sqrt{x + \frac{3}{8}}.$$

Si può dimostrare come questa operazione trasformi dati con statistica di Poisson in dati approssimativamente gaussiani, con valore atteso

$$m = 2\sqrt{\mu + \frac{3}{8}} - \frac{1}{4\sqrt{\mu}}$$

e deviazione standard unitaria. Di conseguenza, è possibile applicare ai dati questa trasformazione, operare una soppressione di rumore gaussiano additivo, e quindi ritornare al dominio originario applicando la trasformazione inversa algebrica[2]:

$$y \to \frac{y^2}{2} - \frac{3}{8}.$$

trama di interferenza costruttiva o distruttiva che risulta in punti chiari o scuri nell'immagine. Questo processo di rumore è moltiplicativo, ed è descritto da una distribuzione di Rayleigh.

[2]L'inversione della trasformata di Anscombe non è scevra da complicazioni di carattere matematico, come l'introduzione di una polarizzazione non desiderata sul valore atteso dei dati. Esistono versioni della trasformata inversa di Anscombe volte a mitigare questo problema, ma per semplicità non verranno prese in considerazione in questo lavoro.

Se il segnale è affetto da un rumore misto gaussiano e di Poisson, esiste una generalizzazione della trasformata di Anscombe adatta a questa situazione [13]. La trasformata di Fisz consegue uno scopo simile [14] .

Metodi di questo tipo, che differiscono per i dettagli dello stadio di stabilizzazione della varianza e per la susseguente eliminazione del rumore gaussiano additivo, sono descritti in [13], [15], [16], [17]. I metodi di stabilizzazione della varianza danno buoni risultati quando l'intensità di segnale è "adeguata" [5]. Se il numero di fotoni incidenti per unità di spazio e di tempo è molto limitato, un metodo diretto, che considera la statistica del rumore di Poisson e adotta un approccio bayesiano che tenga conto di tale statistica, è generalmente considerato più efficace. C'è comunque da aggiungere che il concetto di "intensità adeguata" è molto vago, e che le prestazioni riportate dai vari algoritmi descritti in letteratura sono fortemente legate al tipo di segnale analizzato, e difficilmente generalizzabili.

Metodi basati sulle *wavelet*

Per quanto riguarda l'effettiva soppressione del rumore, preceduta o meno da stabilizzazione della varianza, tra i molti metodi proposti possiamo citare il *filtraggio non locale* (NL) [14], che sfrutta la ridondanza intrinseca nel segnale, misurandolo in opportune aree (*patch*) per ricostruire l'informazione su tutta l'estensione dell'immagine. Si tratta di un filtraggio non lineare, concettualmente simile al filtro mediano. Questo metodo tuttavia ha una scarsa efficienza computazionale. Numerosi metodi sono basati sulla *trasformata wavelet* [18], che generalmente è molto efficace per la soppressione del rumore gaussiano e non [19]. La trasformata *wavelet*, ben nota nell'analisi dei segnali, fornisce una rappresentazione multirisoluzione del segnale, scomponendolo in una serie di versioni a scala differente mediante un banco di filtri numerici. Il concetto di scala sostituisce quindi il più tradizionale concetto di frequenza, e risulta più adatto a catturare le caratteristiche intrinseche di processi complicati come le immagini, dove coesistono strutture a scale molto differenti (per esempio bordi, *texture*, aree relativamente uniformi, ecc.), localizzate sia nel tempo che in frequenza. Ogni versione del segnale a un dato valore di scala può essere elaborata separatamente dalle altre, e trattiene i dettagli del segnale (e del rumore) proporzionali a tale scala. A seconda della statistica del rumore, questo si va a collocare preferibilmente ad alcuni specifici valori di scala, e lo stesso vale per il segnale utile, che normalmente è molto concentrato a valori di scala elevati, corrispondenti a bande di basse frequenze, mentre il rumore non ha questo comportamento. È quindi possibile limitare il rumore preservando in larga misura i dettagli del segnale applicando opportune soglie ai coefficienti dove è massimo il contributo del rumore e minimo quello del segnale utile. Per esempio, si possono applicare soglie determinate sulla base della deviazione standard dei coefficienti di ciascun livello di risoluzione.

La trasformata *wavelet* (di cui la trasformata di Haar è un caso particolare) è stata applicata alla microscopia a fluorescenza per esempio in [20], al caso di rumore misto gaussiano e di Poisson in [21], [22], [23] e in [24] senza operare trasformazioni per la stabilizzazione della varianza. Esiste comunque un'estesa letteratura, e i vari metodi proposti variano per la scelta dei banchi di filtri, per il fatto che le varie sottosequenze siano o meno sottocampionate, ecc.. È opportuno notare che, per quanto esistano algoritmi numerici efficienti per il calcolo della trasformata *wavelet*, come il cosiddetto *lifting scheme* [18], si tratta comunque di un metodo abbastanza complesso dal punto di vista computazionale. Di conseguenza, per quanto molto interessanti, con molteplici campi di applicazione e diversi tipi di finalità, in questo lavoro non saranno presi in considerazione approcci basati sulla trasformata *wavelet*, approcci comunque riservati ad approfondimenti futuri.

Metodi bayesiani

I *metodi bayesiani* si sono rivelati promettenti (anche) per gestire i casi di rumore moltiplicativo. Essi si pongono l'obiettivo di stimare al meglio il segnale utile dal punto di vista probabilistico, partendo da misure rumorose e/o affette da distorsione. Sia il segnale che il rumore sono considerati processi stocastici, e modellati con opportuna densità di probabilità (detta *informazione a priori*) e auto/cross-correlazione[3]. Il problema è formulato in termini di massimizzazione di una *funzione di verosimiglianza* (stima ML, *maximum likelihood*), ovvero una funzione di probabilità condizionata. In altre parole, dati due eventi A e B, la verosimiglianza è una funzione del tipo:

$$\mathcal{L}(A,b) = \alpha P(A|B = b),$$

dove α è una costante di proporzionalità. Per il teorema di Bayes, abbiamo che

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)} \tag{2.1}$$

dove sia $P(A|B)$ che $\frac{P(A|B)}{P(A)}$ sono funzioni di verosimiglianza con diverse costanti di proporzionalità, dato che la seconda espressione è proporzionale alla prima. Ora, identifichiamo con A il "segnale misurato" (affetto da rumore, effettivamente osservabile) e con B il "segnale utile" (non direttamente accessibile in quanto, appunto, affetto da rumore). L'Eq. 2.1 esprime la probabilità che B assuma un determinato valore, dato il valore della variabile misurata. Ciò viene espresso in funzione della distribuzione di probabilità dei dati osservati $P(A)$, della verosimiglianza $P(A|B)$ (queste due riconducibili alla verosimiglianza $\frac{P(A|B)}{P(A)}$), e dall'informazione *a priori*

[3]Nel caso di processi stocastici stazionari, la funzione di autocorrelazione e la densità spettrale di potenza sono legati da una relazione biunivoca e invertibile mediante l'operatore di Fourier [25], e quindi portano la stessa informazione.

$P(B)$, che contiene informazione statistica sul segnale utile (se gaussiano, poissoniano ecc.). I vari metodi bayesiani quindi assumono informazione *a priori* su B e su A, ovvero la distribuzione del segnale utile, del rumore e il modello della relazione tra i due (se additiva, moltiplicativa ecc.). Essi puntano a massimizzare il rapporto di verosimiglianza, ovvero la probabilità che il segnale utile assuma un determinato valore, dato il fatto di avere osservato (misurato) uno o più valori del segnale rumoroso A. Il caso più semplice è quello in cui sia il segnale utile che quello osservato abbiano distribuzione gaussiana. In questa situazione, il problema si può formulare con un sistema di equazioni lineari, che ammettono quindi una soluzione analitica. Se invece queste condizioni non sono soddisfatte, come nel caso di rumore di Poisson o Rayleigh, e/o distribuzione *a priori* non gaussiana, il problema non ammette più una soluzione analitica, e deve essere risolto numericamente mediante algoritmi iterativi. Bisogna osservare che, in quest'ultimo caso, i metodi ML risultano spesso mal posti dal punto di vista algoritmico, nel senso che non offrono soluzione univoca. In questi casi, è necessario introdurre termini di regolarizzazione, in modo da garantire che l'algoritmo di ottimizzazione converga verso un massimo assoluto [26]. Tra i metodi bayesiani proposti nel caso di rumore con statistica di Poisson o di Rayleigh, citiamo per esempio [26], che propone un formalismo che unifica il problema di rumore non additivo di diversa natura usando come approccio la soluzione dell'equazione di Sylvester-Lyanupov. Strettamente collegati sono i metodi cosiddetti *maximum a posteriori*, che inglobano la distribuzione *a priori* nell'obiettivo di ottimizzazione, e possono essere visti come una regolarizzazione dei metodi ML.

Alcuni dei più utili metodi di riduzione di rumore moltiplicativo adottano l'approccio bayesiano su dati trasformati, adottando un modello statistico del segnale osservato, una distribuzione *a priori* e una tecnica di massimizzazione della funzione di verosimiglianza adatti al dominio in cui i dati vengono rappresentati. Si veda per esempio [27] o [28], dove si usano approcci bayesiani su dati trasformati mediante *wavelet* o simili (*ridgelet*, *curvelet* [17]). I metodi bayesiani danno generalente risultati superiori a quelli che si ottengono operando direttamente soglie sui coefficienti della trasformata *wavelet*.

Il filtro di Wiener per serie discrete

Diamo qui una breve descrizione del filtro di Wiener, sia come esempio semplice di stimatore bayesiano, sia perché questo filtro può anche essere applicato per la riduzione del rumore di Poisson, previa opportuna trasformazione (per esempio, mediante trasformata di Anscombe). L'obiettivo principale del filtro di Wiener (anche se non l'unico) è la riduzione del rumore presente in un segnale, che per semplicità verrà considerato monodimensionale; l'estensione a filtri bi-tridimensionali è matematicamente ovvia. Il filtro viene progettato per ottenere l'eliminazione del rumore,

ammesso che questo abbia un'occupazione in frequenza separata da quella del segnale utile. Esso adotta un approccio di tipo bayesiano per conseguire questo risultato, o la sua migliore possibile approssimazione nel caso l'ipotesi di separazione spettrale non sia perfettamente soddisfatta. In altre parole, se c'è una parziale sovrapposizione spettrale tra segnale e rumore, il filtro di Wiener consegue la migliore possibile approssimazione al problema della soppressione del rumore. Quindi, dato un segnale $s[n]$ di cui sono note le caratteristiche spettrali, e un'osservazione $x[n] = s[n] + \eta[n]$, dove $\eta[n]$ è un processo di rumore del quale sono anche note le caratteristiche spettrali, si vuole identificare un filtro lineare e tempo-invariante (*linear, time-invariant*, LTI) la cui uscita sia *la più simile possibile* al segnale $s[n]$, sulla base di una opportuna misura di errore da minimizzare. Le ipotesi che si assumono per la soluzione di questo problema sono le seguenti:

- Il rumore è additivo; infatti, l'osservazione è rappresentata dalla somma del segnale utile e del rumore, entrambi modellati come processi stocastici.

- Sia il segnale che il rumore sono processi stazionari e lineari, con funzioni di autocorrelazione e di mutua correlazione note. Se, come spesso si ipotizza, segnale e rumore sono statisticamente scorrelati, la soluzione del problema risulta assai semplificata. Si noti comunque che questa ipotesi è irrealistica nel caso di rumore di Poisson, anche rielaborato per mezzo della trasformata di Anscombe, che non è conunque in grado di sopprimere la correlazione segnale-rumore.

- La soluzione del problema, ovvero la risposta all'impulso del filtro che si ottiene, deve essere fisicamente realizzabile (causale).

Quindi si cercherà una stima di $s[n]$:

$$\hat{s}[n] = g[n] * \{s[n] + \eta[n]\}$$

dove $g[n]$ rappresenta la risposta all'impulso del filtro da ottimizzare, e $*$ denota prodotto di convoluzione [25]. L'ottimizzazione implica rendere minima una funzione di errore

$$e[n] = s[n + \alpha] - \hat{s}[n]$$

Se $\alpha > 0$ si parla di *filtro di predizione*; se $\alpha = 0$ si parla di *filtro di Wiener* propriamente detto; infine, se $\alpha < 0$ si parla di *filtro di smoothing*. Il dettaglio dell'algoritmo di ottimizzazione esula dalla presente tesi; si veda per esempio [29]. Il filtro di Wiener è comunque disponibile nella maggior parte dei pacchetti software per l'elaborazione di segnali, ad esempio nel *toolbox Signal Processing* di MATLAB.

2.1.2 Miglioramento del segnale

Come discusso in precedenza, i fenomeni che alterano la qualità di un'immagine sono essenzialmente due: il rumore, e lo sfocamento (*blurring*) indotto dalla risoluzione finita dello strumento. È quindi auspicabile lavorare, oltre che sulla riduzione del rumore, anche sulla correzione del segnale stesso, cercando di ovviare allo sfocamento indotto dallo strumento. Questo processo viene detto *deconvoluzione* ed è riconducibile a un filtraggio lineare.

Si ipotizzi di illuminare un oggetto puntiforme, che quindi matematicamente è assimilabile a una delta di Dirac [25]. L'ottica del microscopio induce un allargamento dell'oggetto puntiforme, che pertanto non è più assimilabile a una delta di Dirac ma a una forma d'onda bidimensionale caratteristica dello strumento, detta *point spread function* (PSF). Quindi, la PSF descrive la risposta del sistema a una sorgente puntiforme (risposta all'impulso). Essa di fatto è la "macchia" luminosa nell'immagine che rappresenta un oggetto puntiforme di dimensioni inferiori al limite di risoluzione dello strumento. Dal punto di vista della teoria dei sistemi, la trasformata di Fourier della PSF rappresenta la risposta in frequenza (o funzione di trasferimento) del sistema, ammesso che quest'ultimo sia lineare. Quello di PSF è un concetto molto utile in ottica, astronomia, microscopia ottica ed elettronica. Il grado di sfocamento, esprimibile in termini dell'occupazione spaziale della PSF (la sua "larghezza") è una misura della qualità del sistema di formazione delle immagini. Un esempio di PSF di un microscopio a fluorescenza è riportata in Fig. 2.1.

Nei sistemi non coerenti di formazione delle immagini, come la microscopia a fluorescenza, il processo di formazione delle immagini è lineare in potenza, ovvero: se il sistema illumina contemporaneamente due oggetti A e B, il risultato è pari alla somma delle immagini che si otterrebbero illuminando separatamente A e B. Questo è dovuto al fatto che non c'è interazione tra i fotoni emessi da A e B, per cui il fatto di acquisire un'immagine di A non influisce con l'immagine di B. Di conseguenza, l'immagine di un oggetto complesso può essere interpretata come la convoluzione tra l'oggetto "vero" e la PSF[4]. In altre parole, se si scompone l'oggetto in punti discreti di diversa intensità, l'immagine viene composta come somma della PSF in cui ogni punto viene trasformato. Pertanto, siccome la PSF è una funzione intrinseca del sistema ottico, conoscere le sue proprietà è sufficiente per descrivere l'intera immagine. Siccome il processo è formalizzabile in termini di convoluzione tra il segnale vero (non noto) e la PSF, la conoscenza di questa funzione permette di affrontare il problema inverso, ovvero ricostruire il segnale utile mediante deconvoluzione tra il segnale misurato (l'immagine) e la PSF. Quindi, in ottica il termine "deconvoluzione" indica il processo mediante il quale si inverte la distorsione indotta dallo

[4]Si noti come, qualora la luce rilevata sia coerente, il processo di formazione dell'immagine risulti lineare in campo complesso, per cui, se si registra l'intensità della luce ricevuta, si possono avere effetti di interferenza costruttiva o distruttiva, quindi non lineari.

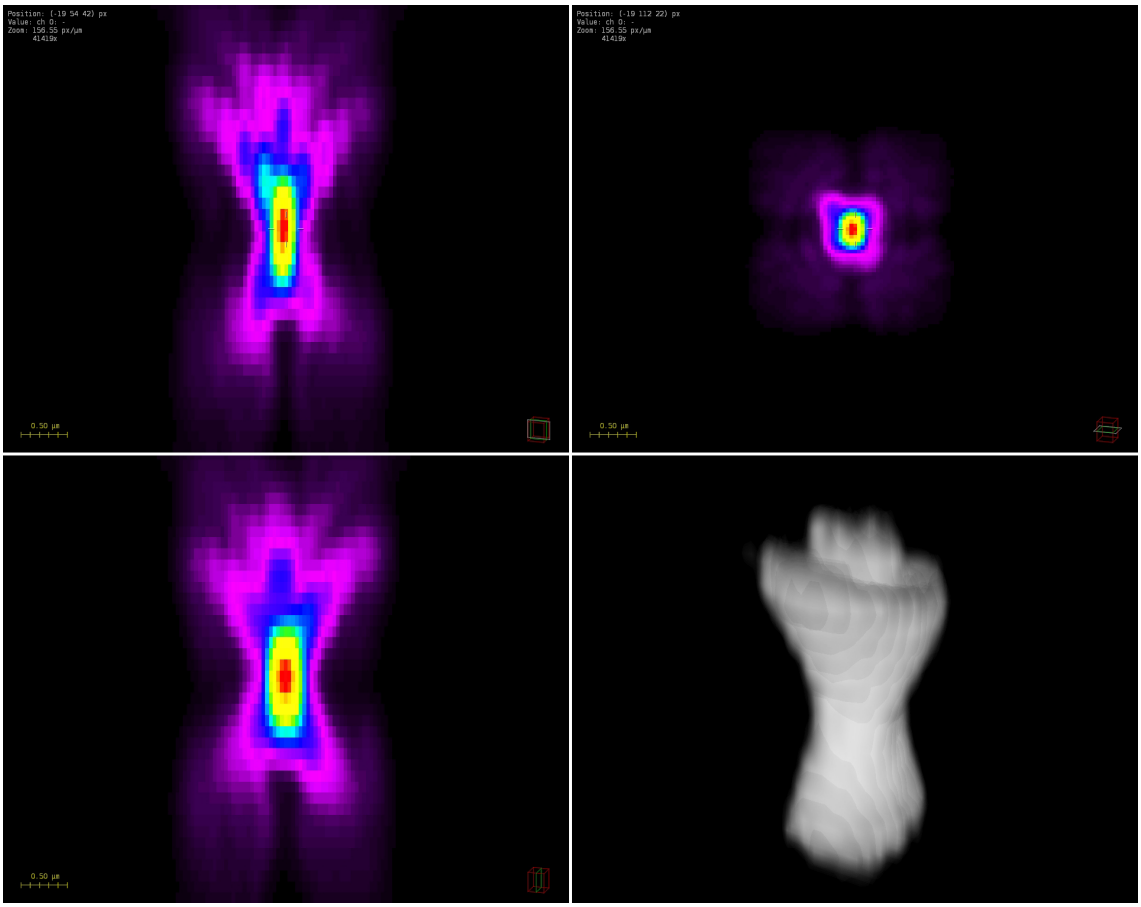


Figura 2.1. Esempio di PSF di microscopio a fluorescenza. Tratta da `en.wikipedia.org`

strumento - microscopio, telescopio ecc. - ottenendo immagini più nitide. La deconvoluzione si effettua nel dominio numerico, con un opportuno software. Si tratta di calcolare l'inversa della PSF, e di effettuare la convoluzione dell'immagine misurata con la PSF inversa ottenendo come risultato, in linea di principio, l'immagine originale non distorta.

Nella pratica, la PSF di uno strumento non è mai perfettamente nota, e inoltre è possibile che uno strumento abbia diverse PSF a seconda dei differenti piani focali o delle posizioni spaziali. Quindi, non è detto che il comportamento dello strumento sia perfettamente lineare. Tuttavia, è possibile stimare la PSF mediante diversi assetti sperimentali. Migliore è la stima della PSF, più accurati sono i risultati della deconvoluzione, anche se al prezzo di maggiore complessità. Nel caso di sorgenti laser, la PSF può essere modellata con funzioni gaussiane. Effettuare la deconvoluzione dell'immagine acquisita con una PSF gaussiana inversa può migliorare la

risoluzione anche di un ordine di grandezza. Il semplice modello gaussiano, anche se ovviamente approssimativo, viene spesso adottato nella pratica, data la possibilità di effettuare la deconvoluzione in modo semplice e veloce.

Un noto algoritmo iterativo di deconvoluzione è quello di Lucy-Richardson, frequentemente utilizzato anche nel campo della microscopia a fluorescenza, anche se sviluppato per applicazioni di astronomia, e adatto al caso di processi poissoniani. Se la PSF non è nota e non è possibile stimarla, si può adottare l'approccio della cosiddetta *deconvoluzione cieca (blind)*, che prevede di provare diverse PSF possibili e stabilire se l'immagine viene migliorata o meno eseguendo la corrispondente deconvoluzione.

Bisogna notare come, anche se concettualmente le operazioni di riduzione del rumore e deconvoluzione sono distinte, esse vengono spesso eseguite insieme; inoltre, una deconvoluzione di per sé, effettuando di fatto un filtraggio, ha come effetto anche una riduzione del rumore. Nella pratica, i metodi più frequentemente usati recentemente nel campo della microscopia a fluorescenza sono basati sulla minimizzazione di un funzionale che è la somma di due termini: uno di *fedeltà dei dati*, che dipende dal processo di formazione dell'immagine e quindi dalla PSF e dal processo di rumore coinvolto, e uno di regolarizzazione, che impone qualche condizione *a priori* alla soluzione, per evitare che un algoritmo iterativo converga verso soluzioni assurde o inaccettabili. Per esempio, si possono imporre condizioni sul fatto che la soluzione (che rappresenta un'energia) sia positiva, scremando quindi risultati spuri che potrebbero essere determinati dalla presenza di rumore, e quindi di fatto operando una riduzione del rumore. Tra i metodi più popolari, citiamo ancora una volta metodi basati sulle *wavelet* [30] o sulla norma cosiddetta *variazione totale* [31]. In [32] si adotta un operatore di tipo laplaciano o hessiano, ovvero una derivata di secondo ordine, per conseguire la soppressione di rumore e insieme la regolarizzazione della norma usata per la deconvoluzione. In ogni caso, effettuando la riduzione di rumore seguita da deconvoluzione, si ottengono risultati notevoli [15]. Quindi, la lezione imparata negli ultimi anni è che bisogna combinare in modo intelligente riduzione di rumore e deconvoluzione, riuscendo in questo modo ad ovviare alla illuminazione estremamente bassa necessaria per evitare fototossicità e *photobleaching* mantenendo al tempo stesso la possibilità di ricostruire dettagli strutturali.

2.1.3 Effettiva identificazione delle comete

Una volta pre-elaborata l'immagine, bisogna procedere all'effettiva identificazione delle particelle luminose. Il limite fondamentale dell'accuratezza della localizzazione è

$$\epsilon = \frac{\sigma}{\sqrt{N}}$$

dove σ è la deviazione standard dell'approssimazione gaussiana della PSF, e N è il numero di fotoni rilevati nell'unità di spazio e di tempo [2]. Questo valore è dell'ordine delle decine di nanometri nei casi pratici, considerando anche i fenomeni di discretizzazione e le varie sorgenti di rumore additivo (non fotonico). Si tratta comunque di valori assai più piccoli dell'estensione della PSF (centinaia di nm) e quindi del limite di risoluzione che consente di separare due oggetti. Quindi, anche dal punto di vista teorico, gli oggetti possono essere localizzati con precisione sotto il limite di risoluzione.

Applicazione di soglie

L'approccio più semplice per ottenere questo risultato consiste nel calcolare i centri di massa (centroidi) dei potenziali oggetti, sottoponendo l'immagine a un operatore di soglia. Di fatto, questo significa ipotizzare che lo sfondo sia statico, e comunque che l'intensità dell'oggetto sia decisamente superiore a quella dello sfondo. Normalmente, si valuta l'istogramma dell'immagine, e lo si usa per decidere il valore di soglia da applicare. Tuttavia, se SNR è basso, questo approccio non dà buoni risultati, in quanto una netta separazione delle intensità tra pixel utili e sfondo non è garantita. Inoltre, tipicamente lo sfondo non è uniforme, e quindi non è corretto usare un unico valore globale di soglia, mentre bisognerebbe usare soglie diverse a seconda delle regioni su cui si opera. Ciò viene proposto in molti lavori, ma ancora le prestazioni normalmente non sono adeguate quando SNR è molto basso. Quindi, l'applicazione di soglie adattative è un approccio corretto, ma generalmente costituisce lo stadio finale di un algoritmo più elaborato.

Sfruttamento di coerenza spaziale

In casi critici di SNR, risulta auspicabile non basarsi sulla semplice applicazione di un operatore di soglia (globale o locale che sia), ma sfruttare la coerenza spaziale tra pixel vicini. Infatti, una macchia che rappresenta un segnale utile deve comunque essere costituita da pixel che hanno una qualche caratteristica (colore, luminosità,...) che li contraddistingue dallo sfondo, e porta a identificare l'insieme di tali pixel come un oggetto (come già accennato, la forma non è un criterio usabile in quanto siamo sotto il limite di risoluzione dello strumento). Quindi, si può eseguire una registrazione locale dell'immagine, ovvero, per ogni oggetto potenzialmente rilevato, usare la distribuzione locale dell'intensità come termine di confronto con oggetti vicini per capire se essi appartengono alla stessa categoria. Bisogna scegliere e ottimizzare una misura di somiglianza, come ad esempio la cross-correlazione normalizzata o la somma delle differenze quadratiche tra i pixel, oppure eseguire una comparazione dell'oggetto con un modello matematico predefinito che rappresenti la distribuzione presunta delle intensità (tipicamente gaussiana). In particolare, nel caso in cui le

comete abbiano forma particolarmente elongata, è noto come l'applicazione di soglie tenda a dare falsi positivi. In questo caso ha senso modellare le comete come oggetti bidimensionali di forma gaussiana (anisotropi in generale). Si usano opportuni filtri per trovare i massimi locali, quindi si cimenta ogni massimo locale con una funzione gaussiana per vedere se c'è accordo con il modello dell'oggetto da rilevare [9]. Ovviamente, l'approccio è probabilistico, ovvero si stima la probabilità che si tratti di un oggetto utile e non si ottiene un semplice risultato on/off come nel caso di applicazione di soglie. Questo metodo, noto come *gaussian fitting*, verrà preso in considerazione nel seguito. Un interessante approccio della stessa tipologia è basato ancora una volta sull'uso della trasformata *wavelet*. Si può verificare che, se i coefficienti di una certa regione spaziale sono relativi a un oggetto significativo, essi risultano correlati nella dimensione scala, mentre ciò non è verificato se i coefficienti sono relativi allo sfondo. Pertanto, si può usare il prodotto di coefficienti corrispondenti alla stessa regione spaziale ma a scale diverse (eventualmente sottoposti a soglie opportune per la riduzione del rumore) come indicatore del fatto che tali coefficienti corrispondano a un oggetto [33]. Un altro operatore simile impiegato per il rilevamento di oggetti su sfondo non uniforme è il cosiddetto filtro *top hat*. Si tratta di un operatore che applica soglie dinamiche, e discrimina gli oggetti per la loro sezione circolare, date opportune informazioni *a priori* sull'intensità che ci si aspetta da un oggetto significativo. Esso esegue medie dei pixel su opportune sezioni circolari, e facendo ciò ottiene di fatto anche una certa soppressione del rumore.

LoG, DoG, algoritmo *watershed*

L'operatore migliore per ottenere una discriminazione degli oggetti dallo sfondo ottenendo nel contempo una riduzione del rumore (anche correlato) è il cosiddetto filtro *Laplacian of Gaussian* (LoG) [34]. Esso opera sottraendo dall'immagine lo sfondo (calcolato su base locale), preservando le strutture che hanno alta probabilità di essere oggetti utili, e contemporaneamente riducendo il rumore. Si tratta di un filtro passabanda, la cui risposta all'impulso è la derivata seconda di una gaussiana (la cosiddetta funzione *mexican hat*). Nella pratica, innanzitutto si filtra l'immagine con un *kernel* gaussiano a una determinata scala t:

$$L(x,y,t) = f(x,y) * g(x,y,t)$$

$$g(x,y,t) = \frac{1}{2\pi t^2} e^{-\frac{x^2+y^2}{2t^2}}$$

dove t è la scala a cui esiste l'oggetto da identificare, ovvero la sua dimensione stimata (si ritiene che l'oggetto sia circolare con raggio $r = t\sqrt{2}$). Quindi, si applica all'immagine filtrata l'operatore laplaciano

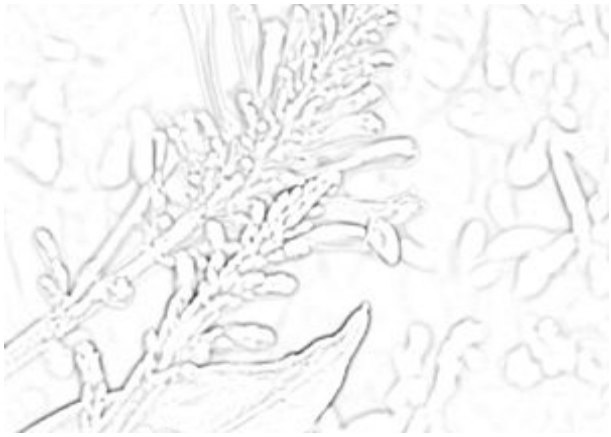
$$\nabla^2 L = L_{xx} + L_{yy} = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

Nell'immagine così elaborata, gli oggetti luminosi risaltano decisamente sullo sfondo (forniscono coefficienti grandi e positivi), mentre gli oggetti molto scuri producono coefficienti grandi e di segno negativo, quindi possono essere anch'essi identificati facilmente. Il risultato del filtraggio LoG viene usato come mappa dei potenziali oggetti, ed è quindi sottoposto a un'operazione di soglia [35]. Un operatore simile è il rilevatore H-dome [36]. Un limite di questi metodi è il fatto di dipendere dalla conoscenza delle dimensioni degli oggetti da evidenziare (parametro t). Una scelta errata di questo parametro porta a un eccesso di falsi positivi o di falsi negativi (oggetti vicini fusi tra loro e interpretati come un oggetto unico). Purtroppo, scegliere il parametro t, e quindi impostare la larghezza di banda del filtro sulla base della conoscenza delle dimensioni degli oggetti da rilevare, non sempre è possibile né realistico, dato che sovente nell'immagine coesistono oggetti di dimensioni diverse. Sono peraltro state proposte versioni dell'algoritmo volte ad alleviare la criticità di questo parametro, per esempio basate su una stima multiscala della presenza di oggetti di dimensioni variabili o non note [35]. Una curiosità: studi di fisiologia della retina sembrano indicare il fatto che la trasmissione dell'informazione visiva ai centri superiori sia preceduta da un'elaborazione dell'informazione che segue un criterio simile al LoG.

Un'approssimazione dell'operatore LoG è il cosiddetto *Difference of Gaussians* (DoG) [9]. Si creano due versioni dell'immagine filtrate passapasso, usando due filtri con *kernel* gaussiano e deviazioni standard rispettivamente σ_1 e $\sigma_2 > \sigma_1$. Quindi, la versione filtrata a banda più stretta viene sottratta alla versione filtrata a banda più larga; l'immagine ottenuta è di fatto filtrata passabanda, in quanto essa contiene solo le frequenze intermedie tra le frequenze di taglio dei due filtri, e presenta sfondo relativamente uniforme. Si dimostra che, se il rapporto tra σ_2 e σ_1 vale circa 2, questa operazione equivale a eseguire la convoluzione dell'immagine originaria con una funzione del tipo *mexican hat*, che di fatto è la derivata seconda di un *kernel* gaussiano. Normalmente il rapporto tra σ_2 e σ_1 viene mantenuto a valori 4:1 oppure 5:1, quindi questa equivalenza non vale in modo esatto. Tuttavia, nella pratica DoG è un modo semplice per realizzare (seppure in modo approssimato) l'operatore LoG. Il *kernel* gaussiano stretto σ_1 elimina le fluttuazioni di alta frequenza dovute al rumore; il *kernel* gaussiano più largo σ_2 elimina variazioni a scala maggiore. Ovviamente, continua a sussistere il problema che le particelle possono variare molto in intensità e grandezza, e anche eventualmente sovrapporsi. Ai fini pratici, DoG (come LoG) consegue riduzione del rumore (che tipicamente occupa una banda superiore a quella del segnale utile) e aumenta la visibilità dei bordi degli oggetti (*edge sharpening*), facilitando le operazioni di confronto con soglie opportune per l'identificazione dei medesimi. Quindi, una volta applicato l'operatore LoG o DoG, ci si ritrova con un'immagine a sfondo uniforme, rumore ridotto, e bordi degli oggetti ben evidenziati, come rappresentato in Fig. 2.2.

Per estrarre da questa immagine le coordinate del centro di massa degli oggetti,

(a)



(b)

Figura 2.2. Immagine originale (a) e sottoposta all'elaborazione *Difference of Gaussians* (b). Da `en.wikipedia.org`

si può applicare l'algoritmo di soglia adattativa noto come *watershed* [9]. Si interpreta l'immagine come un "paesaggio 3D", con gli oggetti che si elevano nettamente dallo sfondo come montagne, e si applicano soglie progressivamente descrescenti che intercettino queste montagne. Si permette a ogni particella di aumentare la propria sezione man mano che la soglia diminuisce, finchè essa si fonde con un'altra particella (e a questo punto si prende atto del fatto che gli oggetti siano due, e questi vengono considerati come separati), oppure si raggiunge il livello minimo m della soglia. La scelta di m va eseguita con attenzione; m dev'essere abbastanza grande da evitare fluttuazioni fittizie dovute a variazioni dello sfondo, ma abbastanza piccolo da permettere di interpretare come distinti due massimi locali dovuti a particelle

molto vicine. I parametri da definire comunque sono tre: i valori di soglia minima e massima, e il passo di variazione progressiva della soglia. Normalmente essi si determinano sulla base della deviazione standard media di un certo numero di immagini centrate intorno a quella che si sta analizzando (in [9] si considerano 5 immagini). Non è invece necessario fare ipotesi su dimensioni, forma e intensità delle particelle.

Anche se non ha pretese di essere ottimo nel caso di rumore non AWGN, DoG è un operatore molto utilizzato in pratica per la sua semplicità, per il fatto di essere tarato mediante due soli parametri (le deviazioni standard dei due *kernel* gaussiani) e per le sue oneste prestazioni pratiche. Tuttavia, mentre l'ipotesi che il rumore si estenda su bande di frequenza superiori a quelle del segnale utile è ragionevole praticamente in ogni caso pratico di rumore additivo anche non gaussiano, la sua veridicità nel caso di rumore moltiplicativo, in particolare di Poisson, non è scontata.

2.2 Inseguimento temporale delle comete

Una volta che le particelle sono state rilevate in tutte le immagini, bisogna stabilire tra di loro una corrispondenza in modo da tracciarne la traiettoria temporale. Questo problema non è affatto banale. Innanzitutto, il numero di particelle rilevate in ciascuna immagine non è costante. Non tutte le particelle vengono trovate e distinte dallo sfondo in ciascuna immagine, alcuni oggetti rilevati sono in realtà effetto dello sfondo o del rumore, alcune particelle possono entrare o uscire dal campo dell'immagine o trovarsi temporaneamente fuori dal piano focale, altre possono avvicinarsi tra loro sotto il limite di risoluzione minimo necessario per interpretarle come separate, una particella in un'immagine può rivelare di essere un ammasso di particelle in un'immagine successiva. A causa del *photobleaching*, l'intensità luminosa della singola particella non è costante lungo la traiettoria temporale da essa percorsa.

Come in ogni problema che vuole trovare una corrispondenza tra oggetti, è necessario definire una misura di distanza, dove con il termine *distanza* non si indica necessariamente o solamente una misura di prossimità spaziale, ma eventualmente anche di similitudine relativamente ad aspetti quali il volume o la superficie occupata, l'intensità o la variazione attesa dell'intensità (velocità della traiettoria in evoluzione) o altri aspetti che siano considerati significativi. Tutte queste grandezze possono essere combinate tra loro e con la prossimità spaziale per ottenere una misura composita che chiameremo distanza, e che racchiude le informazioni ritenute utili per definire l'oggetto di interesse.

I metodi per inseguire le particelle lungo la dimensione temporale si dividono in locali e globali. I metodi locali lavorano particella per particella; una specifica particella in una data immagine è associata a una particella nell'immagine successiva in modo da minimizzare la prescelta metrica di distanza (*nearest neighbor*). Questo approccio è molto usato per la sua semplicità, e fornisce discreti risultati qualora le

particelle siano poche e ben separate. Tuttavia, nei casi pratici di maggiore interesse, le particelle in ciascuna immagine sono dense e poco separate, e i metodi locali di inseguimento forniscono soluzioni ambigue, talvolta assestandosi su minimi relativi della metrica di distanza. In questi casi, che come detto sono la maggioranza di quelli di interesse pratico, è necessario adottare un approccio globale. Si tratta di coinvolgere nella ricerca non le particelle isolate ma tutte quelle rilevate nell'immagine, o almeno quelle che cadono in un opportuno intorno del punto dove si sta lavorando, e puntare alla ricerca di una corrispondenza ottima globale. Molte tecniche di ottimizzazione globale sono state sviluppate per l'elaborazione di immagini generiche [37]. Tuttavia, esse non danno buoni risultati qualora applicate alle immagini biologiche, perché normalmente assumono ipotesi troppo semplicistiche sul movimento delle particelle, non sufficienti ad esempio per descrivere l'eterogeneità del moto dei MT o la coesistenza di particelle che seguono tipologie di traiettorie diverse. La situazione è aggravata dal basso SNR, per cui sono frequenti errori di inseguimento delle particelle anche usando algoritmi sofisticati. Ad esempio, nel campo della videosorveglianza, è molto comune rivelare il moto di un oggetto sottraendo alle immagini lo sfondo, che però nel nostro caso non è di intensità costante. Per contro, nelle immagini biologiche è minore il problema dell'occlusione, problema invece enfatizzato nella videosorveglianza, mentre non è possibile sfruttare la morfologia degli oggetti. In generale, allo stato attuale si può sostenere che non è possibile prescindere da una verifica qualitativa dei risultati forniti da *qualsiasi* algoritmo. Negli ultimi anni sono entrati nell'uso comune tecniche di *rendering* 3D dei dati, che non hanno altra ambizione se non quella di facilitare l'operazione di verifica qualitativa (in altre parole, di aiutare l'occhio fornendo visualizzazioni più intuitive dei dati). Una volta fatta la verifica, è possibile estrarre dai dati parametri quali spostamento, velocità, accelerazione delle singole particelle o valori medi in determinate regioni delle immagini. In genere, è preferibile calcolare medie o istogrammi per ricavare dati stabili da una massa di particelle difficilmente gestibili singolarmente, e ottenere se non altro i parametri dei modelli di movimento dominanti. Tuttavia, si è purtroppo piuttosto lontani dal possedere metodi di analisi sufficientemente robusti, che consentano di ottenere informazioni affidabili sulle traiettorie singole per poter distinguere statisticamente l'eterogeneità di comportamento delle cellule *in vivo*.

Per aumentare la sensitività e la riproducibilità dei risultati forniti da un algoritmo automatico, bisogna capire come gli algoritmi esistenti si confrontino con la percezione nel sistema visivo umano. Questo è sicuramente meno accurato e riproducibile di un algoritmo automatico nell'identificare le coordinate precise di una particella. Ciò nonostante, gli umani sono ancora molto migliori nel confermare la presenza o assenza di una particella in un'immagine, anche e soprattutto a livelli molto bassi di intensità, e a stabilire le corrispondenze tra particelle in immagini successive in caso di alta densità. Si è visto sperimentalmente che a SNR di 1dB, che è così basso da non consentire ad alcun algoritmo esistente di funzionare, dato che i

segnali sono virtualmente indistinguibili dal rumore, gli esseri umani riescono ancora a visualizzare la particella e a fare grossolane stime della sua direzione e velocità. La spiegazione di ciò è psicofisiologica e neurologica. Un aspetto della superiorità umana rispetto alla macchina sta nel fatto che la percezione umana integra informazioni da sottosistemi diversi, sensibili al movimento, alla luminanza, alla *texture*, e pesa in modo diverso aspetti su cui si concentra un'attenzione selettiva. Esiste una rete di rilevatori che si attivano o meno a seconda della direzione del movimento (ovvero, essi si attivano in cascata se sono sintonizzati sulla stessa direzione del movimento), operando come un integratore spazio-temporale sensibile alla direzione. Inoltre, rilevatori di movimento selettivi esistono a molte scale spazio-temporali. L'integrazione temporale è di cruciale importanza, così come le ipotesi sulla coerenza temporale per rilevare il movimento locale di distribuzioni spaziali deboli o ambigue. Ugualmente importante è il processo cognitivo che sta dietro alla percezione umana del movimento. Gli esperti biologi attribuiscono più senso ai dati che osservano rispetto ai non esperti perché sfruttano conoscenze *a priori* sulla dinamica delle particelle che stanno inseguendo. Attualmente, come discusso, gli algoritmi disponibili eseguono una discriminazione piuttosto stretta tra informazione spaziale e temporale. Metodi per rilevare le particelle stimano la loro posizione immagine per immagine, e successivamente affrontano il problema dell'inseguimento temporale, considerando in genere non più di due o tre immagini successive per volta. Non è così che funziona il sistema visivo umano, anche se attualmente non sembrano esserci concrete possibilità di un approccio automatico differente.

Sicuramente, è possibile e necessario usare tutta l'informazione *a priori* disponibile, specialmente per consentire il riconoscimento di segnali deboli. In [9] si enfatizza la necessità di sfruttare la conoscenza del modello di movimento delle particelle. Questo può essere un moto casuale (browniano), come nel caso delle vescicole, o diretto, lineare, come nel caso dei MT, che peraltro presentano anche le fasi di pausa e di *shrinkage*. Non è realistico quindi pensare di avere un algoritmo di inseguimento di particelle che funzioni altrettando bene in queste due situazioni, ma sarà necessario specificare che tipo di movimento si intende inseguire, e integrarne le informazioni nell'algoritmo.

Sempre in [9] si affronta il problema dell'inseguimento delle traiettorie di MT marcati mediante proteine fluorescenti +TIP come EB1 o EB3. Si adotta un algoritmo suddiviso in due passi. Il primo passo esegue il legame delle particelle tra due immagini consecutive. Tale passo segue un approccio temporale *greedy*[5], ma spazialmente è globalmente ottimo, ovvero: si collegano le particelle da un'immagine alla successiva, anche se sarebbe forse migliore una soluzione che collega una

[5]Un algoritmo di ottimizzazione si dice *greedy* - *ingordo* se esso assume come valida la prima soluzione identificata corrispondente ad un minimo della funzione di costo, senza verificare che tale minimo sia assoluto e non relativo.

particella nell'immagine generica i con un'altra presente in un'immagine $j \neq i+1$. Tuttavia, nell'identificare la corrispondenza tra una particella nell'immagine i e una nell'immagine successiva $i+1$, le particelle vengono considerate tutte, e si persegue un ottimo globale. Siccome, nel caso si inseguano proteine +TIP marcate con EB3, è possibile seguire direttamente solo le fasi di crescita del MT, un secondo passo è necessario per collegare segmenti di traiettoria separati da fasi di pausa o *shrinkage*, in cui di fatto la traiettoria scompare per un certo numero di immagini, ma successivamente ricompare e va associata al medesimo MT. Questa fase viene condotta in modo globalmente ottimo sia nella dimensione spaziale che in quella temporale, e punta alla ricostruzione delle traiettorie complete non andando necessariamente a collegare gli spezzoni di traiettoria temporalmente più vicini, bensì quelli che minimizzano una funzione di costo globale. Entrambi i passi sono affrontati risolvendo un sistema di assegnazione lineare (*linear assignment*, LA), con una funzione di costo che comprende il modello dinamico di movimento dei MT favorendo soluzioni maggiormente aderenti a tale modello, e usando il filtro di Kalman [38].

Il filtro di Kalman è un filtro ricorsivo, ben noto nella teoria dei sistemi, che stima lo stato (l'evoluzione) di un sistema dinamico a partire da una serie di misure soggette a rumore, e assumendo un modello stocastico, quindi anch'esso potenzialmente approssimato, della dinamica del segnale utile. Il filtro di Kalman quindi introduce direttamente nel problema due sorgenti di imprecisione: una legata al fatto che si hanno informazioni *a priori* approssimate su quello che ci si aspetta dall'evoluzione dell'oggetto di interesse (informazioni statistiche sul suo movimento e sulla sua velocità), il secondo legato all'imprecisione delle misure effettuate sul sistema in questione, affette da varie sorgenti di rumore. Il filtro di Kalman dà risultati teoricamente ottimi se sia il rumore sia l'errore sul modello sono variabili gaussiane a media nulla e scorrelate tra loro. Tuttavia, le sue prestazioni sono straordinarie anche quando queste ipotesi non sono verificate, e ciò lo rende una delle più grandi invenzioni del secolo, con migliaia di applicazioni in meccanica, aeronautica, geolocalizzazione (GPS) ecc. Rudolf Emil Kalman, nato a Budapest nel 1930 e recentemente scomparso, nel 2008 è stato insignito della *National Medal of Science* dal presidente Obama, per premiare i suoi straordinari contributi all'avanzamento della conoscenza scientifica (Fig. 2.3).

Non è questa la sede dove affrontare una descrizione dettagliata del filtro di Kalman, oggetto tanto semplice e logico come concezione quanto complicato nei dettagli di funzionamento. Il concetto base comunque è quello di partire da una misura dello stato del sistema di interesse. Ogni volta che si dispone di una nuova misura, si valuta la cosiddetta *innovazione*, ovvero la differenza tra la nuova misura e la precedente stima disponibile (funzione di tutte le misure precedenti). L'innovazione è l'informazione aggiuntiva portata dalla nuova misura, e ad essa viene assegnato un peso che tiene conto di quanto essa sia statisticamente affidabile e praticamente utile per migliorare la stima precedente, tenendo conto della statistica del modello, del

Figura 2.3. Rudolf E. Kalman riceve la *National Medal of Science* dal presidente Obama (immagine tratta dal sito `sting.deis.unibo/sting/Img/`)

rumore di misura e della loro importanza relativa. Tutto ciò viene realizzato tramite una sequenza di operazioni matriciali eseguite da un algoritmo numerico [38], che realizzano e eseguono iterativamente uno stadio *predittore* e uno stadio *correttore*, conseguendo una minimizzazione della covarianza dell'errore di stima.

Una considerazione a parte meritano i metodi basati sull'apprendimento, o *machine learning*. Questi algoritmi sono spesso impiegati in applicazioni in cui determinati oggetti devono essere inseguiti a partire da immagini acquisite da telecamere (esempio: controllo della presenza di pedoni in autostrada o nei sistemi di guida assistita dell'automobile). Il vantaggio di questi metodi sta nel fatto che essi non assumono praticamente alcun modello *a priori* sulle caratteristiche degli oggetti da inseguire (forma, movimento ecc.), ma piuttosto le "imparano" da una certa quantità di dati che vengono loro messi a disposizione. Quindi, anche se non si ha un modello sufficientemente dettagliato, oppure se questo non è stazionario, oppure se gli oggetti da inseguire sono di diverse tipologie ma con qualche caratteristica,

anche sfuggente, in comune (esempio: uomo/animale/veicolo/grosso oggetto fermo in una carreggiata autostradale, o che si muove con moto imprevedibile; uomo grasso/magro/alto/basso/bambino ecc.), basta fornire all'algoritmo immagini che appartengano alla tipologia voluta, unitamente alle regole su "che cosa imparare". Questi sistemi hanno ovviamente suscitato l'interesse dei ricercatori anche per l'inseguimento di traiettorie da immagini di microscopia. Tuttavia, dopo un'iniziale proliferazione di tentativi, la conclusione pressoché univoca è stata che questi metodi non sono adatti al caso in esame. Il motivo è che, per tutti i motivi già discussi (inaffidabilità delle informazioni morfologiche, basso SNR, estrema eterogeneità dei dati tra un esperimento e l'altro), lo sforzo necessario per ricavare dati di *ground truth* affidabili e in quantità adeguata per addestrare l'algoritmo è comparabile con lo sforzo necessario per analizzare manualmente i dati. Di conseguenza, questi metodi sono praticamente scomparsi dalle tecniche proposte in letteratura.

2.3 Considerazioni riassuntive

Nei tantissimi algoritmi presentati in letteratura fino ad oggi, il processo di identificazione e inseguimento di particelle in immagini di microscopia a fluorescenza viene suddiviso in fasi di riduzione del rumore, miglioramento del segnale (decorrelazione), rilevamento vero e proprio, inseguimento temporale per la ricostruzione delle traiettorie. In realtà questa classificazione, per quanto utile dal punto di vista concettuale, non è sempre rispettata nelle realizzazioni pratiche, in cui tutte queste operazioni sono profondamente interlacciate, con l'eccezione della fase di inseguimento temporale, effettivamente distinta dalle altre in tutti gli algoritmi considerati. Per esempio, uno stadio di decorrelazione di fatto opera anche la riduzione del rumore, anche se magari non è ottimizzato per questo scopo. LoG non è altro che un filtro passabanda, quindi riduce sicuramente il rumore presente fuori dalla banda del segnale utile, anche se è ottimizzato per il rilevamento delle particelle. Molti algoritmi presentati nella letteratura scientifica pongono l'attenzione su un particolare aspetto del problema (per esempio: la riduzione del rumore di Poisson) e lo sviscerano perseguendo una soluzione teoricamente ottima *per quell'aspetto*. Tuttavia, nella pratica un metodo usabile deve tenere conto di tutte le sfaccettature del problema, e quindi probabilmente perseguire ottimizzazioni di un singolo aspetto non è così determinante per le prestazioni globali dell'algoritmo. Inoltre, per ambire ad ottenere una soluzione teoricamente ottima di un certo problema, è necessario fare assunzioni molto precise sulla natura del fenomeno in esame. Tuttavia, non è affatto scontato che tali assunzioni siano poi rigorosamente rispettate nei casi pratici di dati sperimentali da analizzare, dati che, tra l'altro, spesso esibiscono un comportamento statisticamente non stazionario. Non per niente, più un algoritmo è "ottimo", più le sue prestazioni risultano dipendenti da una serie di parametri critici, difficili da

determinare nella pratica. Spesso questi algoritmi sono validati su immagini sintetiche o su sottoinsiemi molto specifici di immagini reali, e le loro prestazioni non risultano riproducibili se si usano dati differenti. Questo problema è ben chiaro nella comunità scientifica. Nel 2012 è stata organizzata una gara internazionale, in modo da garantire un confronto il più possibile oggettivo tra metodi differenti, in condizioni comuni (usando gli stessi insiemi di dati) e conducendo il confronto sulla base di criteri di valutazione chiaramente specificati (`www.bioimageanalysis.org/track/`) [39]. Hanno partecipato alla gara quattordici gruppi di ricerca, ai quali sono state concesse tre settimane per presentare i risultati conseguiti dai propri algoritmi su dati simulati rappresentativi di tre situazioni sperimentali: particelle in moto browniano (*random walk*), rappresentativo del movimento di vescicole nel citoplasma; particelle in moto diretto, a velocità approssimativamente costante, rappresentativo del trasporto su MT; movimento misto dei due tipi precedenti, con viraggio casuale dall'uno all'altro, rappresentativo dell'esposizione di recettori di membrana o dell'invasione di virus infettanti. Le particelle sono state modellate come marcate da cromofori fluorescenti e illuminate da microscopio a fluorescenza confocale oppure *wide-field*. Sono stati considerati tre livelli di densità delle particelle da rilevare: bassa (circa 100 particelle per campo), media (500 particelle) e alta (1000 particelle), includendo la possibilità che le particelle appaiano o scompaiano casualmente in immagini successive. Infine, sono stati simulati quattro livelli di rumore, ovvero SNR= 1, 2, 4 e 7 dB[6]. Complessivamente sono quindi state predisposte 48 condizioni sperimentali diverse. Si è optato per l'uso di dati simulati in quanto i dati reali non sono generalmente provvisti di *ground truth* affidabile come quella che può essere associata a dati sintetici.

Per quanto riguarda gli algoritmi in gara, essi comprendono, per quanto riguarda il rilevamento, approcci che spaziano dalla semplice operazione di soglia a elaborazione tipo LoG o DoG, mentre per quanto riguarda la ricostruzione delle traiettorie si sono confrontati metodi basati sulla ricerca di un ottimo locale oppure globale, con o senza uso esplicito di modelli di movimento e stima dello stato dinamico del sistema (Kalman). In [39] si possono trovare i dettagli dei 14 algoritmi confrontati e delle loro prestazioni. In generale, si possono trarre le seguenti conclusioni.

Nessuno dei metodi in gara ha ottenuto le migliori prestazioni in tutti gli assetti sperimentali. Alcuni hanno conseguito buoni risultati in molti di tali assetti, denotando il fatto che *in generale* alcuni metodi si comportano meglio di altri, in particolare quelli basati su approcci di ottimizzazione globale e che sfruttano informazione *a priori* affidabile sul tipo di movimento delle particelle. Punto debole spesso evidenziato, specialmente per gli algoritmi più sofisticati, è la forte dipendenza da alcuni parametri, da cui sorge l'invito ai ricercatori a sviluppare metodi più

[6]Nel confronto, è stata adottata questa definizione: SNR=$(I_0 - I_b)/I_b$, dove I_0 e I_b sono rispettivamente il valore di picco dell'oggetto utile e il valore medio dello sfondo.

robusti e che facciano migliore uso dell'informazione *a priori* senza la necessità di tarare manualmente molti parametri.

In ogni caso, non si è autorizzati a sostenere che alcuni metodi siano i migliori *tout court*, ma solamente che *in genere* o *in un buon numero di casi* essi offrano buoni risultati. Inoltre, le prestazioni di tutti i metodi, in tutte le condizioni sperimentali, tracollano quando SNR scende al di sotto di 4 dB. Le conclusioni riportate in [39] sono quindi che nessun metodo si comporta perfettamente in tutti gli assetti sperimentali, che gli utilizzatori devono essere consapevoli che eventuali eccellenti prestazioni riportate in letteratura possono non essere riprodotte nei propri contesti pratici, e che comunque rimane aperta la sfida di indire una competizione su dati reali e non simulati, con *ground truth* più accurata possibile.

Capitolo 3

Il pacchetto software U-track

U-track è un pacchetto software *open source*, che comprende un'interfaccia grafica e diverse funzioni per analizzare dinamicamente particelle in immagini di microscopia a fluorescenza. Questo software è stato identificato come molto utile per condurre il presente lavoro di tesi, in quanto:

- È scritto in linguaggio MATLAB, quindi facilmente comprensibile e modificabile. Si presta ad essere integrato con altri algoritmi MATLAB, a loro volta (relativamente) semplici da realizzare, grazie all'enorme mole di funzioni statistiche, di filtraggio, di stima ecc. disponibili in ambiente MATLAB.

- È predisposto per la lettura di dati scientifici in un grande numero di formati.

- Fornisce un'ottima interfaccia grafica adattata all'elaborazione di immagini di microscopia.

- Fornisce strumenti statistici per la validazione dei risultati, direttamente utilizzabili e/o facilmente integrabili con altre funzioni che si ritengano importanti.

- Fornisce un discreto insieme di algoritmi predisposti per l'analisi di immagini di microscopia a fluorescenza, tra quelli descritti nel capitolo precedente. Come sarà discusso nel seguito, anche soltanto la semplice selezione dei parametri, il test e la validazione di tali funzioni su dati reali fornisce già informazioni rilevanti e non scontate sul problema dell'inseguimento dei MT marcati con proteine +TIP.

Nel seguito, si daranno alcune basilari informazioni su MATLAB, e si descriveranno le funzionalità realizzate nel pacchetto U-track.

3.1 MATLAB: cenni

MATLAB (abbreviazione di *Matrix Laboratory*) è un programma commerciale per il calcolo numerico e l'analisi statistica dei segnali, nonché un vero e proprio metalinguaggio di programmazione, scritto in linguaggio C e creato dalla MathWorks. MATLAB è particolarmente adatto a manipolare matrici, e permette anche di visualizzare funzioni e dati, realizzare algoritmi, creare interfacce utente, e lavorare con altri programmi. Uno strumentario opzionale permette di unire MATLAB con il motore di calcolo simbolico di Maple. MATLAB gode di un'enorme diffusione in ambiente di ricerca universitaria e industriale, grazie alla sua flessibilità, al fatto di rendere disponibili numerosi strumenti a supporto dei più disparati campi di studio applicativi, e al fatto di funzionare su diversi sistemi operativi, tra cui Windows, Mac OS, GNU/Linux e Unix.

MATLAB fu creato alla fine degli anni Settanta da Cleve Moler, preside del Dipartimento di Scienze Informatiche dell'Università del Nuovo Messico, con lo scopo di offrire agli studenti accesso a LINPACK (libreria scritta in linguaggio Fortan per eseguire operazioni di algebra lineare) e ad EISPACK (libreria analoga per il calcolo di autovalori e autovettori) senza che essi dovessero conoscere il Fortran. MATLAB si diffuse velocemente nelle università e nella comunità dei matematici applicati. Jack Little, un ingegnere, conobbe il programma durante una visita a Moler nel 1983. Riconoscendone il potenziale commerciale, egli propose a Moler di riscrivere MATLAB in linguaggio C. L'anno successivo, i due fondarono la MathWorks per continuare il suo sviluppo e per sfruttarne i proventi.

Quindi, MATLAB è:

- un linguaggio di programmazione particolarmente sviluppato per quanto riguarda la gestione ed elaborazione di vettori e matrici. Si tratta di un linguaggio interpretato, ovvero ogni linea di un programma viene letta, interpretata ed eseguita sul momento senza necessità di compilazione.[1]
- Un ambiente di calcolo scientifico con funzioni altamente specializzate.
- Un ambiente grafico.

Un file .m (M-file) è un programma riconoscibile da Matlab. La scrittura di file .m permette di

- Sperimentare un algoritmo, senza dover introdurre una lunga lista di comandi e senza un eccessivo sforzo di programmazione, grazie alle numerose funzioni di calcolo numerico rese disponibili dalla piattaforma MATLAB.

[1] Il codice MATLAB è interpretato, ma è anche possibile compilarlo tramite il Matlab Compiler rendendo così al contempo chiuso il file sorgente e più veloce l'esecuzione.

- Ottenere programmi che possono essere riutilizzati, per esempio cambiando solo i parametri.

- Scambiare programmi con altri utenti e mettere a disposizione le proprie funzioni (come nel caso di U-track).

Come accennato, MATLAB contiene in sé centinaia di funzioni di calcolo numerico, che possono essere native o incluse nei cosiddetti *toolbox* opzionali la cui licenza si acquista a parte. Questo facilita enormemente la programmazione, in quanto i programmi in MATLAB di solito richiamano numerose di queste funzioni disponibili (per esempio: varie operazioni tra matrici, filtri numerici, trasformate, ecc.). Grazie a queste funzioni, scrivere un programma in MATLAB per risolvere un problema che coinvolge calcolo scientifico risulta molto più semplice e veloce che usare linguaggi come C, Java ecc. Per contro, tipicamente l'efficienza computazionale dei programmi MATLAB risulta inferiore. Poiché il codice Matlab viene scritto in file di testo, è immediatamente trasferibile ad altre piattaforme dove MATLAB è installabile (UniX, Mac, Windows).

3.2 U-track

U-track (versione 2.1.3) è un pacchetto software MATLAB per il *tracking* di particelle multiple sviluppato presso il *Danuser Lab - Laboratory for Computational Cell Biology* della *Harvard Medical School, Department of Cell Biology* (`http://lccb.hms.harvard.edu/index.html`). Questo gruppo di ricerca, altamente multidisciplinare, comprende esperti di biologia, biochimica, ingegneria meccanica, biomedica e dei materiali, informatica, matematica, fisica, *computer vision*. Si occupa di studiare i meccanismi in base ai quali segnali meccanici e chimici si integrano nello spazio e nel tempo, per controllare la dinamica del citoscheletro e il traffico di membrana, adottando un approccio sperimentale che imponga minime perturbazioni e sfrutti l'eterogeneità degli stati dinamici della cellula per validare la gerarchia e la cinetica delle cascate di segnalazione intracellulare.

U-track si propone i seguenti obiettivi [40]:

- mettere a disposizione funzioni per eseguire l'identificazione e l'inseguimento di oggetti in immagini di microscopia con alta densità di particelle e moto eterogeneo (quindi, contesti "problematici");

- raccordare traiettorie parziali, interrotte a causa di mancato rilevamento di particelle in alcune immagini (per esempio, nel caso dei MT, a causa di eventi di pausa o *shrinkage*);

- rilevare le situazioni in cui due particelle si fondono tra loro, o al contrario si separano, a causa di fenomeni di occlusione o di effettivi eventi di aggregazione o dissociazione.

La caratteristica tecnica principale di U-track è il fatto che l'inseguimento è formulato in termini di ottimizzazione globale, la cui soluzione identifica gli insiemi di traiettorie complessivamente più probabili. Per prima cosa, si trovano le corrispondenze tra particelle in immagini consecutive, e quindi si legano i segmenti di traiettoria parziali a formare traiettorie complete. Inoltre, il software mette a disposizione funzioni esplicitamente progettate per eseguire il rilevamento della dinamica dei MT tramite proteine marcate +TIP in sequenze di immagini di cellule *in vivo*, come descritto in [9] e [41]. Queste funzioni erano precedentemente disponibili in un *toolbox* denominato plusTipTracker e ora accorpato in U-track.

3.2.1 Interfaccia grafica

U-track fornisce un'interfaccia grafica che consente di selezionare la sequenza di immagini da elaborare. È possibile creare un nuovo database partendo da un file, caricare un database di immagini già elaborato in precedenza, oppure caricare una sequenza di immagini dal server OMERO [42]. Per quanto riguarda file memorizzati sul proprio PC, essi possono essere importati direttamente, a patto che sia stato installato il *Bio-Formats MATLAB toolbox*, predisposto per interfacciarsi con tutti i principali formati in cui vengono generate le immagini da parte di diversi microscopi ottici, e scaricabile liberamente (`www.openmicroscopy.org/site/products/bio-formats`). Una volta caricati i dati, l'interfaccia prevede che siano introdotte varie informazioni (se non già automaticamente ottenute dai metadati delle immagini da parte di Bio-Formats): *imaging mode* (confocale, TIRF, *wide-field*), tipo di fluoroforo utilizzato, lunghezze d'onda di eccitazione e di emissione, tempo di esposizione, dimensione del pixel in nm, *frame rate* in s^{-1} (numero di immagini al secondo), apertura numerica della lente del microscopio, numero di bit usati dalla camera per codificare un campione (es.: 8 bit implica 256 livelli di grigio). Altre informazioni libere che si ritengano necessarie possono essere aggiunte, e vengono associate alla directory dove si memorizzeranno tutti i dati relativi all'esperimento. L'interfaccia grafica di U-track consente anche di selezionare il tipo di particelle che si vogliono tracciare. Le opzioni sono *particelle singole*, *microtubuli*, *nuclei*. Nel seguito, si farà riferimento esclusivamente all'opzione *microtubuli*, ma è giusto evidenziare che U-track mette anche a disposizione funzioni per inseguire altri tipi di particelle. Infine, una volta selezionata la tipologia di particelle, U-track apre un'ulteriore interfaccia grafica dalla quale è possibile: tarare i parametri specifici delle procedure disponibili per quella tipologia di particelle; lanciare il software per effettuare l'elaborazione; visualizzare i risultati, sia in forma

grafica che come variabili disponibili nel *workspace* di MATLAB ed elaborabili a piacimento. Nel caso in cui l'utente intenda scrivere altre funzioni .m per realizzare elaborazioni di tipo diverso, queste possono essere facilmente integrate con il software esistente e nella suddetta interfaccia grafica, seguendo una serie di opportune regole sintattiche. Pertanto, U-track costituisce un utile strumento anche semplicemente per gestire in modo ordinato le proprie funzioni di elaborazione, sfruttando le potenzialità grafiche di MATLAB per mezzo dell'interfaccia pre-costituita.

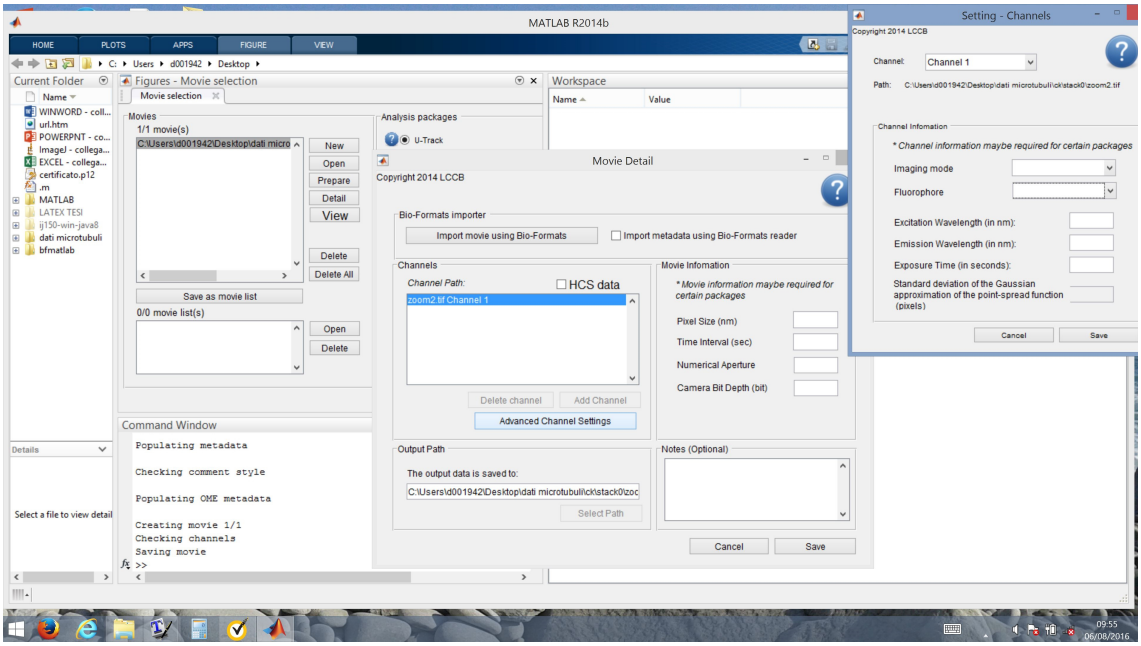


Figura 3.1. *Screenshot* dell'interfaccia grafica principale di U-track

3.2.2 Identificazione dei MT: le procedure disponibili

Il software di elaborazione di U-track, sia che si vogliano tracciare MT che particelle singole o nuclei, segue l'approccio tradizionale, ovvero mantiene separate le funzioni di rilevamento delle particelle nelle singole immagini e l'inseguimento delle traiettorie nel tempo (*tracking*). A ciò si aggiunge un ulteriore livello di elaborazione, ovvero la riclassificazione delle traiettorie identificate e l'analisi dei risultati ottenuti, in cui sono messe a disposizione molte funzioni statistiche per valutare parametri quali le velocità medie di crescita dei MT, le durate medie delle fasi di pausa o *shrinkage* e molte altre. Per quanto riguarda l'analisi della dinamica dei MT, si considera che gli oggetti del tracciamento siano proteine +TIP marcate come EB1 o EB3. Pertanto, gli oggetti da rilevare avranno il tipico aspetto di comete. Il rilevamento di queste

comete viene effettuato scegliendo uno tra due metodi: segmentazione *watershed* e *Gaussian fitting anisotropo*.

Segmentazione *watershed*

Questo algoritmo rappresenta la funzione suggerita per il rilevamento delle comete in ciascuna singola immagine. Innanzitutto, queste vengono sottoposte a un'elaborazione DoG (*Difference of Gaussian*, si veda il Cap. 2). In seguito, per estrarre le coordinate delle particelle dall'immagine in uscita dallo stadio DoG, si usa l'approccio *watershed* (anch'esso descritto nel Cap. 2). Sebbene l'algoritmo non richieda informazioni relative alle dimensioni, alla forma o all'intensità delle comete da rilevare, esso dipende da una serie di parametri, descritti in Tab. 3.1.

Parametro	**Definizione**	**Valore di default**
σ_1	deviazione standard primo kernel gaussiano	1
σ_2	deviazione standard secondo kernel gaussiano	4
s	passo di variazione della soglia del *watershed*	(*)
m	soglia minima di *watershed*	$m = K \cdot s$, $K = 3$
Note:		
(*): calcolato automaticamente in base all'intensità locale delle immagini		
$\sigma_1 > 1$ e/o $K < 3$ aumentano la probabilità di rilevamento se SNR è basso		
$\sigma_2 > 4$ e/o $K > 3$ diminuiscono la probabilità di rilevazioni spurie (falso allarme)		

Tabella 3.1. Parametri della funzione di segmentazione *watershed*

Gaussian fitting anisotropo

Questa funzione viene resa disponibile per quei casi in cui la procedura di segmentazione *watershed* genera tassi eccessivi di false rilevazioni, e ciò tipicamente avviene qualora le code delle comete siano particolarmente elongate. In questo caso, U-track mette a disposizione un algoritmo che si basa su un modello implicito delle comete come forme d'onda gaussiane, anisotrope nelle due dimensioni spaziali. Innanzitutto, le potenziali comete sono identificate per mezzo di opportuni filtri adattabili, quindi si calcolano i massimi locali delle immagini sottoposte a questi fitri. I filtri sono detti adattabili in quanto la loro funzione di trasferimento viene stabilita automaticamente in funzione della deviazione standard dell'approssimazione gaussiana della PSF del microscopio (quest'ultima, calcolata in funzione della lunghezza d'onda di emissione, della dimensione del pixel e dell'apertura numerica della lente). Successivamente, ogni massimo locale identificato viene confrontato con una forma

d'onda gaussiana anisotropa, e si usa un test statistico per identificare con buona probabilità le effettive comete. Il software permette di stabilire l'intervallo di confidenza richiesto per considerare valida una cometa (principale parametro da fissare in questa procedura), e fornisce la lista delle comete identificate, la stima della loro posizione, ampiezza, eccentricità, e la deviazione standard della stima di ciascuno di questi parametri.

3.2.3 Inseguimento delle comete: la procedura disponibile

U-track definisce *traiettoria parziale* il percorso di una estremità *plus* del MT effettivamente inseguita nella sequenza di immagini, e *traiettoria completa* (o traiettoria *tout court*) un insieme di traiettorie parziali che l'algoritmo ritiene opportuno considerare come una traccia unica. Questa distinzione è necessaria, dal momento che solamente le fasi di crescita del MT possono effettivamente essere seguite qualora si usino proteine +TIP marcate con EB1 o EB3. Pertanto, se il MT entra in una fase di pausa o di *shrinkage*, questo non viene più rilevato per un certo numero di immagini (un *gap*). Successivamente, il MT può ritornare in una fase di crescita ed essere quindi nuovamente rilevato. In un primo passo, l'algoritmo si occupa di rilevare le traiettorie parziali. Successivamente, esso impiega un'opportuna funzione di costo per valutare statisticamente quali traiettorie parziali debbano essere combinate, in quanto caratterizzate da un'elevata probabilità di essere relative allo stesso MT. Altre situazioni in cui una traiettoria può essere temporaneamente "persa di vista" e successivamente ritrovata (dando quindi origine, nella prima fase dell'algoritmo, a due traiettorie parziali) sono: che si sia verificato un evento di falso negativo in una certa immagine per una certa particella (quindi, localmente non rilevata per errore) o che una cometa si sia temporaneamente spostata al di fuori del piano focale dello strumento, per poi rientrarvi.

Il primo passo dell'algoritmo è *greedy* temporalmente, ma globalmente ottimo nelle dimensioni spaziali, come anche discusso nel Cap. 2 e in [40]. In sintesi, esso determina l'insieme migliore possibile di corrispondenze tra particelle in due immagini consecutive, tenendo tutte le particelle in considerazione, ma limitandosi per l'appunto a due immagini successive. Se, qualora si stia esaminando l'immagine i-esima, una certa traccia parziale non trova una particella corrispondente nell'immagine $i + 1$ che rispetti i criteri prestabiliti (funzione di costo), la traccia viene ivi sospesa, e memorizzata come traiettoria parziale. Un filtro di Kalman (si veda Cap. 2) viene usato per predire dove ciascuna particella debba trovarsi nell'immagine $i + 1$, dato il suo comportamento nelle immagini i-esima e precedenti. Quindi, l'algoritmo ricostruisce la traiettoria del MT esaminando particelle candidate che si trovano in un opportuno raggio di ricerca intorno alla posizione predetta dal filtro di Kalman. Il raggio di ricerca per ciascuna particella viene stabilito automaticamente basandosi sulle variazioni delle caratteristiche del moto di ciascuna particella. Esso

è piccolo se da parecchio tempo la particella si sta muovendo a velocità costante, il che significa che la predizione fornita dal filtro di Kalman è affidabile. Esso diventa grande se la particella mostra un comportamento alternante tra moto rapido e lento, il che riflette una bassa affidabilità della stima di posizione.

Il secondo passo dell'algoritmo considera le traiettorie parziali ottenute dal primo passo, e le combina in traiettorie complete, rispettando opportuni criteri spazio-temporali. Ciò è condotto in modo globalmente ottimo (sia spazialmente che temporalmente), ovvero si seleziona la migliore possibile costituzione delle traiettorie complete dall'insieme di quelle parziali, tenendo conto di tutte le traiettorie parziali disponibili.

Ambedue i passi sono affrontati risolvendo un problema di assegnazione lineare, in modo da minimizzare una determinata funzione di costo. La funzione di costo per la generazione delle traiettorie parziali (primo passo) tiene conto di opportuni modelli che definiscono la dinamica dei MT [9]. Per quanto riguarda il secondo passo, si considera che i *gap* che separano le traiettorie parziali candidate all'unione possano:

- proiettare la traiettoria in avanti. Detti in questo caso *fgap* - *forward gap*, essi sono dovuti a scomparsa temporanea della cometa dal piano focale, falso negativo di rilevamento, o MT che è entrato in fase di pausa.

- Proiettare la traiettoria all'indietro, per fluttuazioni termiche della posizione del terminale *plus* del MT durante una pausa, catastrofe, ovvero transizione da una fase di pausa a una di *shrinkage*, seguita da una fase di recupero. Questi sono detti *bgap* - *backward gap*.

Una funzione di costo opportuna deve considerare la possibilità che si sia verificato un *fgap* o un *bgap*, senza polarizzarsi su nessuna delle due situazioni. Per ogni traiettoria parziale i, si definisce una regione di ricerca, definita dall'unione di una piccola circonferenza e di due coni. La circonferenza è centrata sull'ultima posizione rilevata della traiettoria parziale, e ha raggio r_{fluct} legato all'incertezza della posizione rilevata e alla fluttuazione termica della posizione dell'estremità *plus* (tipicamente 1 o 2 pixel). I due coni invece si estendono in avanti, parallelamente al vettore della velocità media misurata sulle ultime posizioni della traiettoria parziale, e indietro, seguendo la traiettoria parziale che può talvolta essere curvilinea [9]. I raggi dei due coni sono proporzionali allo spostamento atteso del MT durante il *gap*, data la distribuzione nota della velocità di crescita del MT e il rapporto tra velocità di *shrinkage* e velocità di crescita, valutato sperimentalmente (in genere, la velocità di *shrinkage* è superiore a quella di crescita). Ogni traiettoria parziale j che nasce nella regione di ricerca così definita, entro un numero massimo di immagini pari a Δt_{gap} (vedi sotto), viene considerata una potenziale candidata a essere associata

alla traiettoria parziale i; candidate che originano nella circonferenza o nel cono che proietta in avanti generano *fgap*, altrimenti generano *bgap*.

La funzione che rappresenta il costo di associare la traiettoria parziale i con la traiettoria parziale j è stata scelta empiricamente, e vale:

$$C_{ij} = 1.1^{\Delta t_{gap}} \cdot (d_p^* + d_o^* + [1 - \cos(\theta)])$$

dove:

- Δt_{gap} è la durata del *gap*, espressa come il numero di immagini che intercorre tra la fine della traiettoria parziale i e l'inizio della traiettoria parziale j, e il valore 1.1 è scelto empiricamente;
- d_p e d_o sono due misure di distanza tra le due traiettorie parziali, proiettate nella direzione parallela e ortogonale alla tangente alle traiettorie stesse (si veda [9] per dettagli) e $*$ denota un opportuno operatore di normalizzazione, necessario per bilanciare il peso complessivo delle varie grandezze che costituiscono la funzione di costo;
- θ è l'angolo tra la direzione della tangente alla traiettoria parziale i e la tangente alla traiettoria parziale j.

Questa funzione di costo privilegia il legame tra traiettorie parziali che siano:

- separate da *gap* non eccessivamente lunghi;
- non eccessivamente distanti spazialmente, nelle due direzioni ortogonale e parallela;
- proiettate approssimativamente nella stessa direzione (piccolo angolo θ).

I parametri sensibili dello stadio di ricostruzione delle traiettorie sono riassunti in Tab. 3.2.

Parametro	**Definizione**	**Valore di default**
Δt_{gap}	massima durata del gap, in numero di immagini	12
$MSTL$	minima lunghezza delle traiettorie parziali	3
FA	angolo del cono *forward*	$\pm(25° - 45°)$
BA	angolo del cono *backward*	$\pm(10° - 15°)$
r_{fluct}	raggio di possibile fluttuazione statistica +TIP	1-2 pixel

Tabella 3.2. Parametri della funzione di ricostruzione delle traiettorie

Capitolo 4

I dati sperimentali e gli algoritmi considerati

In questa tesi ci si propone di affrontare il problema dell'inseguimento di MT astrali in immagini di microscopia a fluorescenza in casi pratici, con dati ottenuti da esperimenti condotti presso il Laboratorio. I MT sono marcati nella loro estremità *plus* con proteine +TIP fluorescenti EB3-dtTomato, e le immagini sono ottenute da microscopio confocale Leica TCS SP5-AOBS a 5 canali (Leica Microsystems), come descritto nel Cap. 1.

Scopo della tesi non è tanto identificare algoritmi completamente nuovi per l'analisi di questi dati, quanto piuttosto orientarsi nella massa di algoritmi disponibili in letteratura e selezionare quelli più affidabili per il caso in esame. Infatti, come già discusso in precedenza, l'ambizione di identificare un algoritmo che dia prestazioni ottime in ogni contesto è irrealistica. Inoltre, buona parte dei risultati presentati in letteratura sono ottenuti operando su dati sintetici o su sottoinsiemi molto controllati di dati reali, e ciò è giustificato anche dalla necessità di avere a disposizione una *ground truth* affidabile. Questo problema è risultato evidente anche in questo lavoro. Data anche la limitazione dell'arco temporale in cui si svolge la tesi, per ora ci si limiterà a valutare le prestazioni degli algoritmi prevalentemente confrontando il numero di comete rilevate con *il valore medio* del numero di comete tracciate manualmente dal personale del Laboratorio, unitamente al valore medio di altri parametri di interesse, quali la velocità di crescita, la durata e la lunghezza delle traiettorie. Per quanto parziale e subottimo, questo confronto ha il vantaggio di fornire risultati sintetici, che consentono di effettuare una prima selezione degli algoritmi e dei loro parametri. Inoltre, i summenzionati parametri hanno un diretto interesse come risultato degli esperimenti. Il confronto delle prestazioni con la *ground truth* fornita immagine per immagine dal personale del Laboratorio sarà oggetto di futuri sviluppi.

In questo capitolo, si descriveranno innanzitutto i dati su cui si conducono gli

esperimenti, e alcune misurazioni statistiche svolte per orientare la scelta dei metodi di elaborazione. Si elencheranno quindi gli algoritmi che si è deciso di selezionare, motivando la scelta sulla base della preventiva analisi dei dati. Infine, si descriveranno le metriche usate per validare gli algoritmi e confrontarli tra loro.

4.1 Descrizione dei dati sperimentali

I dati sperimentali sono divisi in due sottoinsiemi: dati *di test* e dati *di controllo*. Ambedue gli insiemi sono ottenuti sperimentalmente con il microscopio confocale su cellule *in vivo* (si veda il Cap. 1 e [1]). Queste immagini sono state selezionate, tra quelle fornite dai tanti esperimenti, innanzitutto perché sono di interesse pratico, e poi perché sono giudicate *difficili* dai ricercatori del Laboratorio, gravate da incertezza nel procedimento di rilevamento manuale delle comete. Ciascuno di questi sottoinsiemi è formato da immagini multiple di formato .TIFF, acquisite in sequenza temporale (*stack*). Esse sono caratterizzate, con alcune eccezioni che saranno rimarcate ove necessario, da questi parametri:

- Risoluzione del quantizzatore: 8 bit per campione, 256 livelli di grigio.
- Tipo di variabili del pixel: unint8 (interi senza segno, 0-255).
- Dimensioni delle immagini: 256x256 pixel.
- Numero di immagini dello *stack* temporale: 120.
- Lunghezze d'onda di eccitazione e di emissione: 561 e 581 nm rispettivamente.
- Dimensioni del pixel: 160.2 nm sia in altezza che in larghezza.
- Dimensioni dello *stack*: 7.5 MB.
- Risoluzione: 6.2431 pixel per μm.
- *Frame rate*: 2 fps (*frame* per secondo).

Dati di test

I dati di test sono sette *stack*, denominati rispettivamente (con riferimento all'esperimento effettuato) *stack 0, 6, 11, 14, 20, 27, 30*, ottenuti con colture cellulari trattate con CITK siRNA e marcate con la proteina fluorescente +TIP EB3-dtTomato. Le caratteristiche sono quelle elencate, con l'eccezione dello *stack 11*, che presenta dimensione del pixel pari a 182.3 nm sia in altezza che in larghezza. In Fig. 4.1 sono riportate le prime immagini di quattro *stack*, ovvero 6, 11, 14, 20. Si può notare come le immagini siano problematiche per una serie di motivi.

1. L'immagine risulta saturata dalla fluorescenza proveniente dai MT che costituiscono il fuso mitotico. Questi ultimi, peraltro, non risultano distinguibili, e sono visualizzati esclusivamente come una larga frazione centrale della cellula che satura verso il colore bianco.

2. Nel presente lavoro, ci si pone lo scopo di seguire i MT astrali e non quelli del fuso; di conseguenza, gli *oggetti di interesse* sono "comete" luminose che si dipartono dalle due estremità del fuso e si dirigono verso la periferia cellulare. Si tratta chiaramente di oggetti al di sotto del limite di diffrazione dello strumento.

3. Lo sfondo è rumoroso, e le comete che rappresentano gli oggetti di interesse sono difficilmente distinguibili da macchie luminose spurie dello sfondo.

4. Anche se questo effetto è apprezzabile solamente seguendo l'evoluzione temporale dell'intero *stack* e non da un'immagine singola, il *photobleaching* è evidente, specialmente in alcuni *stack*, in cui le ultime immagini sono state definite praticamente non utilizzabili dai ricercatori del Laboratorio.

Dati di controllo

I dati di controllo sono sei *stack*, denominati rispettivamente (con riferimento all'esperimento effettuato) *crtl 11, 14, 24, 29, 32, 40*, ottenuti anch'essi marcando l'estremità *plus* del MT con EB3-dtTomato. Le caratteristiche sono quelle elencate, con l'eccezione di *ctrl 11*, che presenta dimensione del pixel pari a 154.5 nm sia in altezza che in larghezza e inoltre consta di 175 immagini, e di *ctrl 32*, che presenta dimensione del pixel pari a 158.7 nm sia in altezza che in larghezza. In Fig. 4.2 sono riportate le prime immagini di quattro *stack* di controllo, ovvero 11, 14, 24 e 40. Si può notare come queste immagini siano più leggibili delle precedenti (soprattutto la 11 e la 14, meno la 24 e la 40), essenzialmente perché la zona interna alla cellula, corrispondente al fuso mitotico è meno saturata, e le estremità *plus* dei MT astrali sono più facilmente distinguibili dallo sfondo, presumibilmente per un minore tasso di rumore. Resta percepibile il fenomeno del *photobleaching*, e ovviamente gli oggetti di interesse sono comunque al di sotto del limite di diffrazione dello strumento.

4.2 Analisi statistica dei dati

4.2.1 Misure di SNR

Innanzitutto, sono state condotte misure di rapporto segnale-rumore (SNR) sui dati a disposizione. SNR ammette varie definizioni. È ben noto dalla teoria dei segnali,

e in particolare dall'elaborazione di immagini, come i valori numerici di SNR varino sensibilmente a seconda della definizione adottata, e non sempre, se si adottano due definizioni diverse, la gerarchia di SNR di un insieme di immagini venga rispettata. Inoltre, sovente non è neppure chiaro che cosa veramente debba essere interpretato come *segnale* e che cosa come *rumore* o comunque disturbo. Questo problema di definizione è molto acuto nel caso sotto esame. Per esempio, nelle nostre immagini il centro delle cellule in mitosi risulta saturato verso l'alto a causa dell'estrema densità di MT. Tuttavia, se quello che si vuole evidenziare è la presenza di MT astrali, l'identificazione di questi ultimi sicuramente viene *disturbata* dal fenomeno di saturazione nella regione del fuso, per cui è molto discutibile se questo debba essere interpretato come segnale o come rumore (disturbo). Un altro esempio: in talune immagini compaiono più cellule, di cui magari una sola in mitosi (*utile*); tuttavia, la presenza delle altre cellule altera la distribuzione dello sfondo dell'immagine, che quindi esibirà un comportamento anomalo. D'altra parte, non è concettualmente esatto considerare questi dati come rumore vero e proprio.

Si è quindi deciso di adottare due misure di SNR:
PSNR o *rapporto segnale-rumore di picco*, così definito:

$$PSNR = 10 \log_{10} \frac{P^2}{MSE}$$

dove P è il valore di picco assunto dai pixel (pari a 255 se la codifica è su 8 bit) e MSE è l'errore quadratico medio tra l'immagine sotto esame e un'immagine di riferimento, dove il segnale è assente e il rumore (e lo sfondo) sono analoghi statisticamente a quelli presenti nell'immagine utile. Nel nostro caso, come immagine di riferimento sono state prese aree dell'immagine facenti parte dello sfondo, ma escludenti la zona centrale saturata contenente il fuso mitotico. PSNR è la metrica di qualità più usata per le immagini generiche, anche se è ben noto che, se si classificano diverse immagini usando PSNR come metrica, la gerarchia che si ottiene non sempre è coerente con una valutazione soggettiva di qualità [37]. Ciò si può spiegare considerando che PSNR è una metrica che usa MSE per valutare il rumore. Questo, di fatto, è proporzionale alla potenza media del rumore, e quindi risulta poco sensibile a fenomeni (quali per esempio la presenza di un sottile bordo spurio, o una sottile quadrettatura) che dal punto di vista della potenza media hanno scarsissimo impatto, mentre sono immediatamente notati e classificati come oggetti spuri, innaturali, sgradevoli (*artifact*) da un osservatore umano. Quindi, questa metrica non corrisponde con esattezza ai criteri di valutazione implicitamente adottati dall'occhio umano.

Mentre il significato pratico e la gravità della presenza di un *artifact* può essere intuito nel caso in cui si stiano elaborando immagini mediche come ecografie, RX, TC (peraltro, casi nei quali PSNR viene raramente usato come metrica, e solo eventualmente per effettuare una prima scrematura degli algoritmi), ciò non è

altrettanto immediato nel caso delle immagini di microscopia. Per acquisire un minimo di sensibilità su questo punto, si è quindi deciso di valutare PSNR per i dati a disposizione.

SNR. Una metrica di SNR alla quale ci si riferisce spesso esplicitamente nel caso delle immagini di microscopia è la seguente:

$$SNR = 10 \log_{10} \frac{I_0 - I_b}{I_b}$$

dove I_0 e I_b sono rispettivamente il valore di picco dell'oggetto utile e il valore medio dello sfondo. Questa è la definizione comunemente usata nei lavori discussi nel Cap. 2.

In Tab. 4.1 sono riportati i valori di SNR e PSNR medi ottenuti per ciascuno degli *stack* a disposizione. Si possono fare le seguenti considerazioni.

Stack	**SNR (dB)**	**PSNR (dB)**
Dati di test		
stack 0	9.5	13.2
stack 6	6.3	11.0
stack 11	5.7	9.4
stack 14	6.6	10.1
stack 20	4.6	11.9
stack 27	5.8	11.7
stack 30	6.2	10.3
Dati di controllo		
ctrl 11	14.0	27.0
ctrl 14	14.0	21.0
ctrl 24	1.8	6.7
ctrl 29	8.5	13.5
ctrl 32	7.4	11.2
ctrl 40	5.7	10.9

Tabella 4.1. Rapporto segnale-rumore e rapporto segnale-rumore di picco dei dati sotto esame

- Anche se numericamente forniscono valori molto diversi tra loro, le due metriche danno approssimativamente misure congruenti: al crescere dell'una cresce anche l'altra metrica, a valori simili dell'una corrispondono valori simili dell'altra. Si tratta comunque di una valutazione semplicistica, in quanto, come discusso, le due metriche non rispecchiano esattamente le stesse caratteristiche dell'immagine, per cui non si può pensare che esse siano del tutto intercambiabili. Il fatto che esse concordino approssimativamente indica la correttezza di massima della distinzione tra segnale utile e rumore.

- La metrica SNR dà valori numericamente simili a quelli pubblicati, relativi a immagini della stessa tipologia di quelle qui considerate, per cui verrà adottata nel seguito, dando la possibilità di confrontarsi con i dati riportati in letteratura.

- Nei dati di test, SNR varia da un minimo di 4.6 a un massimo di 9 dB (media: 6.3 dB). Nei dati di controllo, si nota un comportamento anomalo dello *stack* "ctrl 24"; escludendo questo *stack*, si ottengono valori di SNR decisamente più alti (valore medio: 9.9 dB), il che è il linea con il fatto che questi dati siano considerati *facili* dal personale del Laboratorio. Per quanto riguarda invece lo *stack* anomalo, la cui prima immagine è riportata in Fig. 4.2(c), si può notare come in questa immagine siano visualizzabili quattro cellule, di cui una sola in mitosi. Questo non è l'unico caso in cui ciò accade, ma è uno dei più vistosi. Evidentemente la presenza di cellule spurie, che in questa immagine sono particolarmente visibili, altera pesantemente la statistica dei pixel di sfondo, tanto da fornire valori molto bassi di SNR, non corrispondenti all'effettiva *difficoltà* dell'immagine. In ogni caso, quando si eseguiranno gli algoritmi di inseguimento dei MT astrali, si lavorerà sulla base della varianza locale dei pixel, escludendo di fatto che situazioni come questa abbiano un grosso impatto sulle prestazioni degli algoritmi stessi.

- Quando allo *stack* "ctrl 40", che presenta valori di SNR non lontani da quelli dei dati di test, effettivamente anche da un'analisi qualitativa le immagini di questo *stack* non risultano soggettivamente molto diverse da quelle degli *stack* di test (Fig. 4.2(d)).

- Si noti infine come tutti gli articoli presentati in letteratura siano concordi nel porre a SNR=4 dB il valore di soglia, al di sotto del quale nessun algoritmo finora identificato è in grado di fornire prestazioni soddisfacenti. Alcune delle immagini di test si avvicinano a questo valore limite.

I numeri in Tab. 4.1 sono ottenuti mediando i valori di SNR riportati da ciascuna immagine dello *stack*. A questo punto, ci si è chiesti qual è l'impatto del *photobleaching* su questa metrica, ovvero come varia SNR in funzione dell'indice temporale dello *stack*. I risultati riassuntivi sono riportati in Tab.4.2, in termini dei valori massimi e minimi di SNR e della loro differenza Δ-SNR.

Come prima considerazione, si può notare come in linea di massima le variazioni di SNR lungo la dimensione temporale non siano così significative, il che denota o un effetto non molto evidente del *photobleaching*, o, ipotesi più realistica, il fatto che il *photobleaching* stesso non abbia un grosso impatto su questa metrica. Una spiegazione potrebbe essere legata al fatto che gli oggetti fluorescenti sono piccoli

Stack	**SNRmin (dB)**	**SNRmax (dB)**	**Δ-SNR**	**Andamento**
Dati di test				
stack 0	7.8	10.6	2.8	decrescente
stack 6	6.0	6.6	0.6	oscillante
stack 11	5.5	5.9	0.4	n.s.
stack 14	6.1	7.0	0.9	oscillante
stack 20	4.5	4.8	0.3	n.s.
stack 27	5.5	6.5	1.0	oscillante
stack 30	5.9	6.5	0.6	oscillante
Dati di controllo				
ctrl 11	12.0	14.9	2.9	crescente
ctrl 14	13.8	14.3	0.5	descrescente
ctrl 24	1.6	2.1	0.5	descrescente
ctrl 29	8.1	8.9	0.8	descrescente
ctrl 32	7.1	7.7	0.6	oscillante
ctrl 40	5.4	6.1	0.7	descrescente

Tabella 4.2. Andamento di SNR lungo la dimensione temporale. Variazioni al di sotto di 0.5 dB sono considerate non significative (n.s.)

rispetto allo sfondo, mentre la zona del fuso mitotico (dove il *photobleaching* darebbe effetti vistosi) è stata eliminata dal computo di SNR. In generale, SNR ha un comportamento oscillante di scarso rilievo, anche se in alcuni casi si può osservare un andamento decisamente decrescente, e, in un singolo caso, crescente con il tempo (i casi più significativi sono rappresentati in Fig. 4.3).

Per capire meglio che cosa succede se si include l'intera immagine, compresa la zona del fuso mitotico, per la valutazione del rapporto segnale-rumore, il calcolo di SNR (minimo, massimo e medio) è stato ripetuto considerando appunto l'intera immagine come sfondo. I risultati sono in Tab. 4.3. Si può osservare come i valori ottenuti oscillino pochissimo, e siano comunque molto bassi, spesso inferiori al valore limite di 4 dB. Se ne deduce che:

- È corretto non considerare la zona centrale della cellula, ovvero il fuso mitotico che satura la scala dell'intensità della fluorescenza, nel computo di SNR.

- È altresì vero che qualunque elaborazione si voglia effettuare su MT astrali dovrà essere condotta su regioni dell'immagine che non comprendano l'area del fuso mitotico, priva di interesse pratico e responsabile di un tracollo di SNR, o comunque dovrà basarsi su una stima locale della varianza dei pixel dell'immagine.

Stack	**SNRmin (dB)**	**SNRmax (dB)**	**SNRave (dB)**
Dati di test			
stack 0	5.2	5.3	5.3
stack 6	3.6	3.7	3.7
stack 11	3.1	3.1	3.1
stack 20	4.9	5.1	5.0
stack 27	4.6	4.8	4.7
stack 30	3.3	3.4	3.3
Dati di controllo			
ctrl 11	10.8	14.5	13.4
ctrl 14	9.3	9.4	9.3
ctrl 24	1.6	1.7	1.7
ctrl 29	5.3	5.5	5.4
ctrl 32	3.7	3.8	3.8
ctrl 40	3.6	3.8	3.7

Tabella 4.3. Andamento di SNR minimo (SNRmin), massimo (SNRmax) e medio (SNRave) lungo la dimensione temporale, considerando anche la regione del fuso mitotico

4.2.2 Analisi spettrale

Un'analisi dello spettro bidimensionale di ciascuna delle immagini che compongono i vari *stack* è stata condotta per valutare, almeno qualitativamente, la natura statistica del processo di rumore che colpisce queste immagini. Infatti, se naturalmente non è possibile da questo tipo di indagine avere informazioni relative alla densità di probabilità del processo di rumore, si può invece apprezzare l'occupazione di banda del rumore rispetto a quella del segnale utile (che è tipicamente di natura "passabasso"), e in particolare osservare se lo spettro rimane approssimativamente costante (rumore bianco) o mostra un decadimento all'aumentare della frequenza (rumore "$1/f$"). Queste due fondamentali categorie di processi rumorosi vanno infatti gestite in modo diverso. Si noti come sia il rumore di Poisson che il rumore gaussiano possano appartenere alla categoria del rumore bianco.

In Fig. 4.4 sono riportati gli spettri di potenza normalizzati bidimensionali e una loro sezione monodimensionale, di un'immagine di test e una di controllo. Si è comunque verificato che tutte le immagini di tutti gli *stack* esibiscono un comportamento spettrale qualitativamente simile, come anche succede al variare dell'indice temporale all'interno del singolo *stack*. Si nota un picco localizzato alle basse frequenze, che corriponde al segnale utile, affogato in una base circa costante (al di là delle fluttuazioni statistiche). Ciò denota il fatto che ci si trova di fronte a un classico processo di rumore bianco, che come tale dovrà essere trattato.

4.3 Selezione ragionata degli algoritmi di rilevamento

Abbiamo cercato di orientarci nella grande mole di algoritmi proposti per l'analisi delle immagini di microscopia a fluorescenza, identificando quelli potenzialmente adatti all'analisi dei MT con dati della tipologia di quelli a nostra disposizione. Nella selezione abbiamo seguito questi criteri (in ordine di importanza):

1. Algoritmo adatto all'inseguimento di MT con +TIP marcate con EB1 o EB3.
2. Algoritmo della classe di cui sopra, adatto a operare in contesto di basso SNR.
3. Algoritmo le cui prestazioni non dipendono troppo pesantemente dalla scelta di parametri, o da un numero eccessivo di parametri.
4. Algoritmo di cui esiste codice MATLAB disponibile *open source* (anche per poter impostare eventuali confronti con risultati pubblicati senza che questi possano dipendere da diversità nella realizzazione dell'algoritmo).

Si noti come non ci si limiterà a prendere in considerazione algoritmi completi descritti in letteratura, ma si studierà anche la possibilità di aggiungere stadi o combinare algoritmi diversi, cercando quindi di lavorare su tutte le dimensioni del problema descritte nel Cap. 2: soppressione del rumore, decorrelazione, identificazione vera e propria.

4.3.1 Segmentazione *watershed*

Questo metodo rappresenta certamente la prima scelta per quanto riguarda gli algoritmi di *comet detection*. Esso soddisfa il punto 1 della lista dei requisiti che guidano la selezione degli algoritmi, in quanto proposto e validato in [9] proprio per l'applicazione in questione. Si ricorda che questo algoritmo comprende uno stadio di filtraggio con *Difference of Gaussians* (DoG), ottenuto sottraendo tra di loro due versioni dell'immagine filtrate con due *kernel* gaussiani a diversa deviazione standard, seguito dal vero e proprio stadio di rilevamento basato sull'applicazione di soglie adattative. Siccome il rumore presente nelle immagini sotto esame è decisamente a larga banda rispetto al segnale utile, e inoltre SNR è basso, usare un filtro passabanda per far risaltare i bordi degli oggetti ha senso, in quanto così facendo si evita di enfatizzare il rumore di alta frequenza. DoG è noto per essere adatto all'elaborazione di immagini molto rumorose (punto 2 della lista dei criteri per la selezione degli algoritmi). Esso approssima anche l'operatore *Laplacian of Gaussian* (LoG) quando il rapporto tra le deviazioni standard dei due *kernel* gaussiani è vicino a 2. Valori pratici di questo rapporto in applicazioni di rilevamento di oggetti sono

comunque 4:1 o 5:1, e in questo secondo caso DoG approssima bene i campi recettivi delle cellule gangliari della retina. Esso è semplice da realizzare ed è presente nel *toolbox* MATLAB U-track. Esso dipende da tre parametri: le deviazioni standard dei due *kernel* gaussiani, e il passo di variazione della soglia per la segmentazione *watershed* (si veda Cap. 2). Uno dei risultati di questo lavoro sarà proprio analizzare l'impatto della scelta di questi parametri sulle prestazioni dell'algoritmo.

4.3.2 *Model fitting* gaussiano anisotropo

In [9], questo algoritmo viene proposto per risolvere i casi in cui le estremità +TIP marcate dei MT creino comete particolarmente elongate; in questa situazione, lo stesso articolo sostiene che il *watershed* risulti affetto da un tasso troppo elevato di falsi positivi. Anche se, da un'analisi qualitativa delle immagini sotto esame, non sembra che nel nostro caso le comete siano particolarmente elongate, in un confronto di prestazioni sembra corretto includere anche questo metodo, che soddisfa comunque buona parte dei criteri di scelta. Esso esegue una ricerca degli oggetti di interesse (le comete) assumendo che esse si adattino a una forma gaussiana anisotropa bidimensionale. Prima si esegue un filtraggio adattativo assumendo PSF gaussiana, e si rileva un'approssimativa posizione dei picchi di intensità. In un intorno di questi picchi si esegue il *fitting* con la forma d'onda gaussiana anisotropa, e si procede al rilevamento della presenza della cometa adottando un approccio bayesiano. I parametri da fissare sono: un valore di attendibilità per i test statistici (parametro critico, da cui dipende il tasso di falsi positivi e falsi negativi), la dimensione della forma d'onda anisotropa per effettuare il *fitting*, e la minima distanza tra due oggetti rilevati sufficiente per poterli considerare separati.

4.3.3 Trasformata Anscombe-Wiener

La trasformata di Anscombe è utile per trasformare dati affetti da rumore con statistica di Poisson in dati con rumore additivo di statistica gaussiana. Si è quindi deciso di realizzare questa trasformata, e ovviamente la sua inversa, per integrarla in algoritmi che prevedano un trattamento preferenziale del rumore additivo gaussiano. Per quest'ultimo passo si è deciso di usare il filtro di Wiener, ottimo per rumore AWGN. Si tratta di un filtraggio adattativo che risulta più selettivo di un filtro lineare confrontabile, e quindi preserva meglio i bordi e i contenuti di alta frequenza del segnale. Sembra intessante operare una stima ottimale dell'immagine di microscopia usando il filtro di Wiener dopo la trasformata di Anscombe per operare una riduzione del rumore, anche se non ci sono di fatto garanzie teoriche di buon funzionamento in quanto l'ipotesi di rumore AWGN non è verificata con esattezza.

4.3.4 Trasformata LOG-Wiener

È ben noto come applicando un operatore logaritmico a un segnale affetto da rumore moltiplicativo, il rumore si trasformi sì in un processo additivo, ma statisticamente non assimilabile a rumore additivo gaussiano bianco. Questo è vero sia per il rumore *speckle* che per il processo di Poisson. Tuttavia, nel primo caso, si è visto come l'uso di una tecnica di soppressione di rumore adatta al caso AWGN, sebbene non fondata teoricamente, possa comunque dare risultati soddisfacenti in molti casi. Invece, nel caso di processo di Poisson, non sono stati trovati in letteratura esempi di applicazione di questo metodo. Di conseguenza, pur essendo pienamente consapevoli che non esiste alcun fondamento teorico a favore di questo, si è deciso di provare ad applicare ai dati una trasformazione logaritmica seguita da un filtro di Wiener (*LOG-Wiener*) e quindi dall'operazione inversa (elevamento a potenza). La giustificazione di questo tentativo sta nella semplicità del metodo, e nella non chiarissima distribuzione statistica del processo in esame, il che porta a tentare di svincolarsi da eccessivi limiti teorici.

4.3.5 Metodi di deconvoluzione

Come già ripetutamente detto, l'immagine vera[1] è deteriorata dal rumore e dalla distorsione indotta dalla PSF della lente. Operare una deconvoluzione significa applicare l'operatore inverso a quello di convoluzione con la PSF, e quindi annullare gli effetti di quest'ultimo. È quindi fondamentale, per ottenere buoni risultati in termini di miglioramento della qualità del segnale, conoscere bene la PSF, cosa non sempre possibile. In questa tesi, intendiamo validare la deconvoluzione con metodo di Lucy-Richardson, che esegue in modo iterativo un'approssimazione della deconvoluzione, usando tecniche di ottimizzazione adatte a una statistica di Poisson, e non richiede che siano fatte particolari ipotesi su eventuale ulteriore rumore additivo. Inoltre, sembra ragionevole anche validare l'algoritmo di deconvoluzione cieca, che non richiede la conoscenza della PSF (a parte una stima iniziale, peraltro determinante per la velocità di convergenza dell'algoritmo), la quale è essa stessa oggetto della stima operata iterativamente in base al criterio della massima verosimiglianza.

4.4 L'algoritmo di inseguimento

Per quanto riguarda l'algoritmo di inseguimento temporale delle comete, si è deciso di adottare l'algoritmo *+TIP comet tracking* in [9]. Infatti:

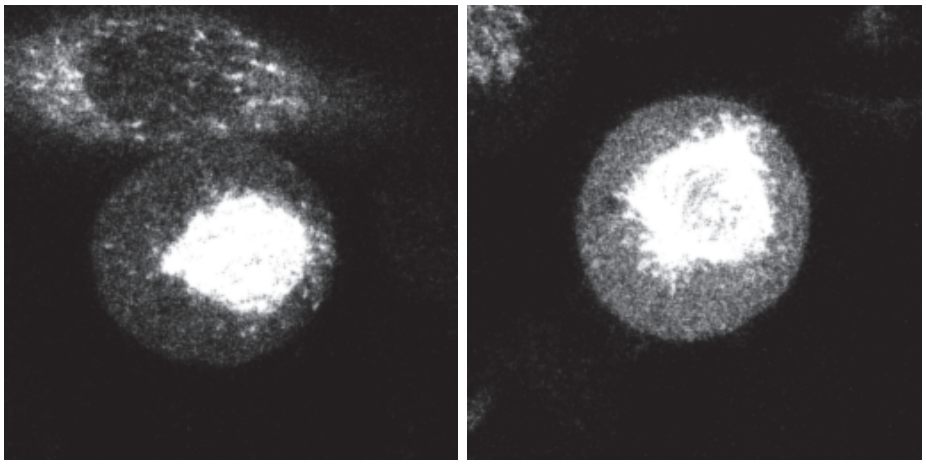
[1] Si noti come l'immagine "vera" sia un'astrazione, e in realtà non esista. Esiste solo l'immagine deteriorata, e quella "vera" è definita come l'immagine che si avrebbe se il sistema ottico fosse ideale, e non ci fosse alcuna sorgente di rumore.

- Esso fa uso dell'informazione *a priori* sul modello di movimento degli oggetti di rilevare, e questo è universalmente riconosciuto come indispensabile per conseguire buoni risultati.
- Esso usa un filtro di Kalman per stimare in modo ottimo lo stato dinamico del sistema (ovvero, dove ci si aspetta di trovare con maggiore probabilità la cometa nell'immagine $i + 1$, dato che essa è stata rilevata nell'immagine i).
- Esso risolve un problema di assegnazione lineare, rilevando le traiettorie parziali e connettendole in modo globalmente ottimo assumento un opportuno modello delle fasi di pausa e *shrinkage*.
- Esso adotta un'opportuna funzione di costo, le cui variabili e i relativi pesi sono facili da cambiare, anche se in questa tesi si è deciso di non modificare la funzione di costo, essendo questa già studiata per risolvere il problema che si sta affrontando. Modifiche della funzione di costo sono quindi lasciate agli sviluppi futuri.
- Esso è disponibile all'interno del pacchetto U-track.

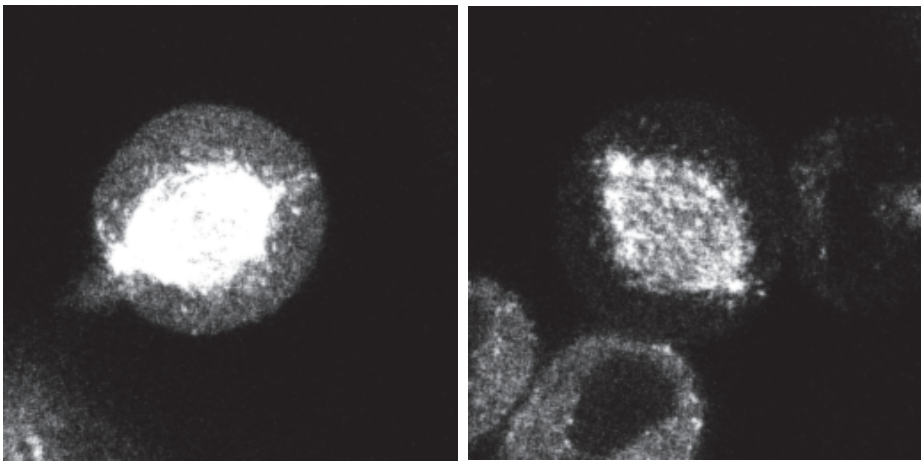
Non sono stati presi in considerazione metodi basati sull'apprendimento (*machine learning*) per i motivi discussi nel Cap. 2.

4.5 Misure effettuate per la validazione

Il primo stadio di rilevamento sarà validato sulla base del numero medio di comete identificate nelle immagini di test e di controllo, confrontato con l'analogo numero ottenuto manualmente. Questo porterà alla selezione dell'algoritmo con la migliore aderenza *media* ai risultati ottenuti dall'analisi manuale delle immagini, relativamente alla selettività nel passo di rilevamento delle comete. Una volta inglobato anche lo stadio di inseguimento delle traiettorie, sull'algoritmo selezionato saranno effettuate prove più approfondite, volte a determinare anche la velocità, la durata e la lunghezza media delle traiettorie complete e delle varie fasi. Ancora una volta, si intende operare un confronto con i dati ottenuti manualmente in termini di informazione media relativa al singolo *stack*. Un confronto diretto, traiettoria per traiettoria, dei risultati forniti dall'algoritmo selezionato con la *ground truth* ottenuta dall'analisi manuale è riservato ai possibili sviluppi futuri.

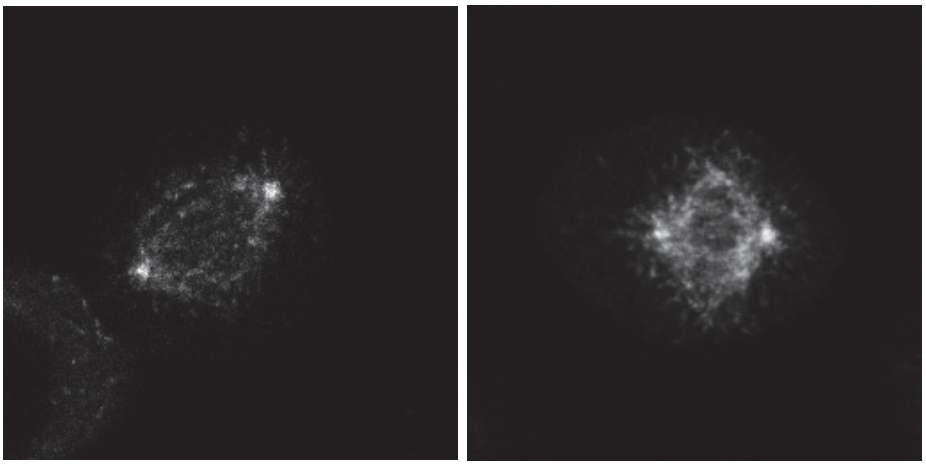


(a) (b)

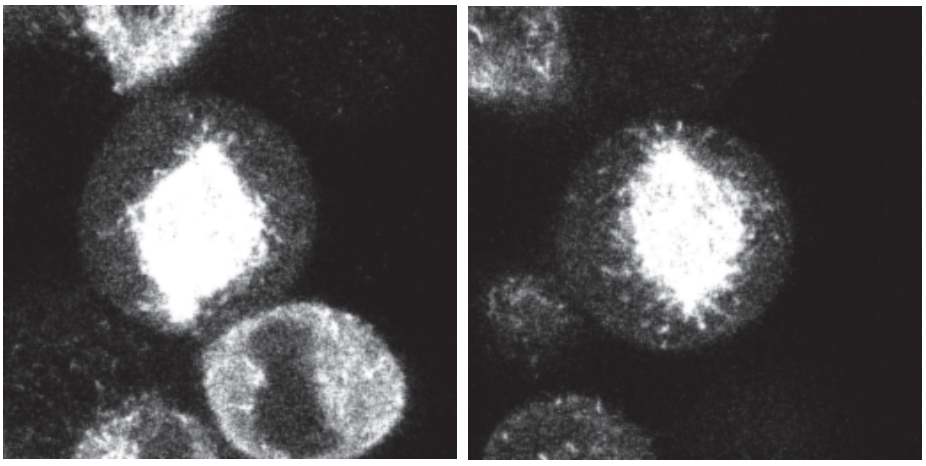


(c) (d)

Figura 4.1. Esempio di immagini di test. (a): *stack* 6; (b): *stack* 11; (c): *stack* 14; (d): *stack* 20



(a) (b)



(c) (d)

Figura 4.2. Esempio di immagini di controllo. (a): *ctrl* 11; (b): *ctrl* 14; (c): *ctrl* 24; (d): *ctrl* 40

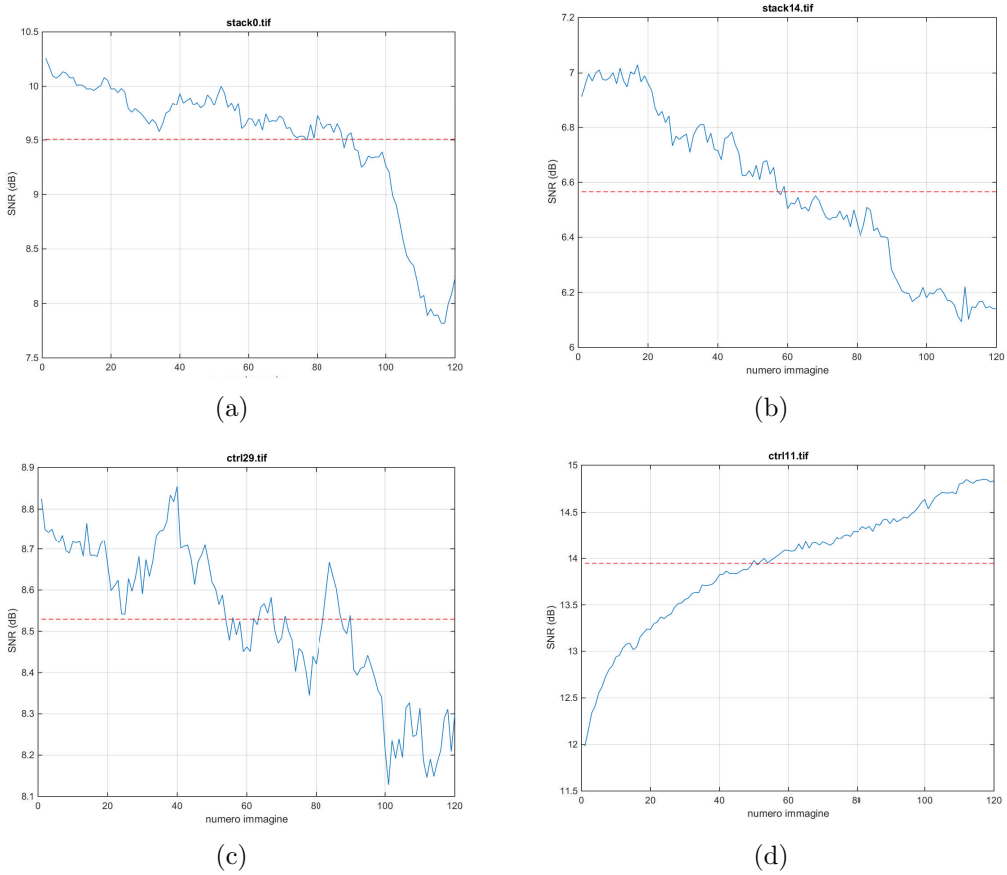


Figura 4.3. Andamento di SNR in funzione del numero dell'immagine. (a): *stack* 0; (b): *stack* 14; (c): *ctrl* 29; (d): *ctrl* 11

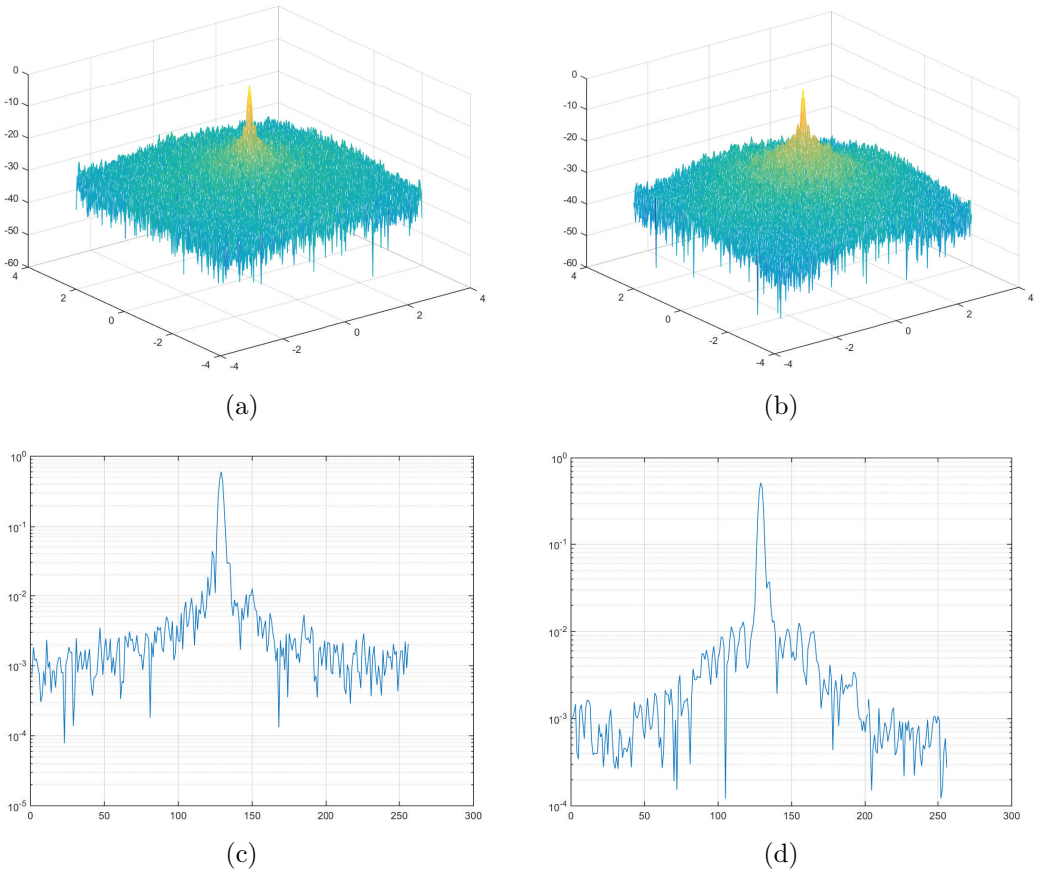


Figura 4.4. Spettro bidimensionale (modulo). (a): *stack* 0; (b): *ctrl* 14. Spettro monodimensionale di una riga dell'immagine. (c): *stack* 0; (d): *crtl* 14

Capitolo 5

Risultati

5.1 Algoritmo *watershed*

Questo algoritmo è disponibile nel pacchetto U-track, e realizza prima di tutto il filtraggio DoG e successivamente lo stadio *watershed* vero e proprio. Esso contiene un certo numero di parametri; di conseguenza, la prima cosa che si è voluta verificare è come la scelta di tali parametri condizioni il numero di comete rilevate dall'algoritmo. Riassumiamo qui di seguito i principali parametri, e i loro valori di *default* così come discussi in [9] e stabiliti in U-track.

- Deviazione standard del primo *kernel* gaussiano: $\sigma_1 = 1$. Indicazioni: aumentare se SNR è molto basso per migliorare la probabilità di rilevamento.
- Deviazione standard del secondo *kernel* gaussiano: $\sigma_2 = 4$. Indicazioni: aumentare per ridurre il tasso di falsi positivi (FPR).
- Passo di variazione della soglia per il *watershed*: s. Calcolato automaticamente sulla base della deviazione standard locale dell'immagine (dinamica locale dei coefficienti).
- Soglia minima per il *watershed*: $m = K \cdot s$, $K = 3$. Indicazioni: diminuire K se SNR è molto basso per migliorare la probabilità di rilevamento. Aumentare K per ridurre FPR.

5.1.1 Selezione dei parametri

Per valutare la sensibilità dell'algoritmo rispetto alla scelta dei parametri, e per fissare i valori dei medesimi che siano il più possibile ottimali per la tipologia dei dati a disposizione, si è deciso di lavorare con i dati di controllo (*stack* "ctrl") e si è seguita la seguente procedura.

- Da una serie di esperimenti preliminari effettuati su varie immagini, sia di test che di controllo, con i valori di *default* dei parametri, si è notato come il numero di comete rilevate sia estremamente alto, se confrontato con il numero medio di comete rilevate manualmente sugli stessi dati dai ricercatori del Laboratorio. Evidentemente si tratta di una situazione in cui i risultati sono affetti da un elevato FPR.

- Non ha senso quindi elevare σ_1, che viene tenuto fermo al suo valore di *default* $\sigma_1 = 1$.

- Per quanto riguarda σ_2, aumentarne il valore al di sopra di quello di *default* $\sigma_2 = 4$ può aiutare a ridurre FPR. In letteratura è peraltro riportato il fatto che il rapporto tra σ_2 e σ_1 debba tipicamente essere 4:1 o 5:1. Pertanto, in queste prove si è deciso di non allontanarsi esageratamente da questi valori. Si eseguiranno prove con $\sigma_2 = 4,5,6,8$, quest'ultimo prescelto per validare la sensibilità dell'algoritmo a valori molto grandi di σ_2 .

- Per quanto riguarda K, elevarlo al di sopra del valore di *default* $K = 3$ riduce FPR. Si è quindi deciso di ricavare risultati con i valori $K = 3,4,5,6$.

- Usiamo tutta la cellula per validare la scelta dei parametri. Infatti, l'elaborazione dell'algoritmo *watershed* è su base locale, e quindi non dovrebbe essere influenzata dalla saturazione indotta dal fuso mitotico (regione in cui comunque l'algoritmo non dovrebbe identificare alcuna cometa).

Come accennato in precedenza, si ritiene che l'elaborazione sia affetta da alto FPR in quanto il numero medio delle comete rilevate oltrepassa sensibilmente il numero medio di comete rilevate manualmente in questi stessi *stack*. Vengono riportati in media 40 MT nucleati per minuto per centrosoma nelle cellule di controllo. Siccome un minuto corrisponde alla durata dell'intero *stack*, se si vedono nucleare 40 MT al minuto significa che mediamente in ogni immagine se ne vede un numero confrontabile con questo (quando un MT nasce, esso è visibile per un certo numero di immagini prima di scomparire, ma nel frattempo ne saranno nucleati altri). Le percentuali di FP e FN si possono valutare come segue [3]:

$$FP = \left(1 - \frac{M}{D}\right)\%$$

e

$$FN = \left(1 - \frac{M}{G}\right)\%$$

dove G è il numero di comete rilevate manualmente, D il numero di comete rilevate dall'algoritmo, M il *match* tra i due. Anche se, per stimare in modo preciso queste

grandezze, bisognerebbe valutarle immagine per immagine sulla base della *ground truth* a disposizione, in questa tesi si esegue solamente una valutazione media e molto approssimata, con l'unico fine di orientarsi tra le varie opzioni di parametri dell'algoritmo usando una metrica coerente. Poniamo quindi $M = 40$, il che significa ipotizzare che $G \approx M$ e $FN \approx 0$. Questa ipotesi sembra giustificata in quanto, come si è visto, l'algoritmo rileva tantissime comete, molte di più di quelle rilevate manualmente, quindi la situazione è dominata da FP in eccesso e non da eventuali (poco numerosi) FN. Nella scelta dei parametri quindi si dovrà ambire a ottenere determinati FPR. Per avere $FP = x\%$ sarà necessario che

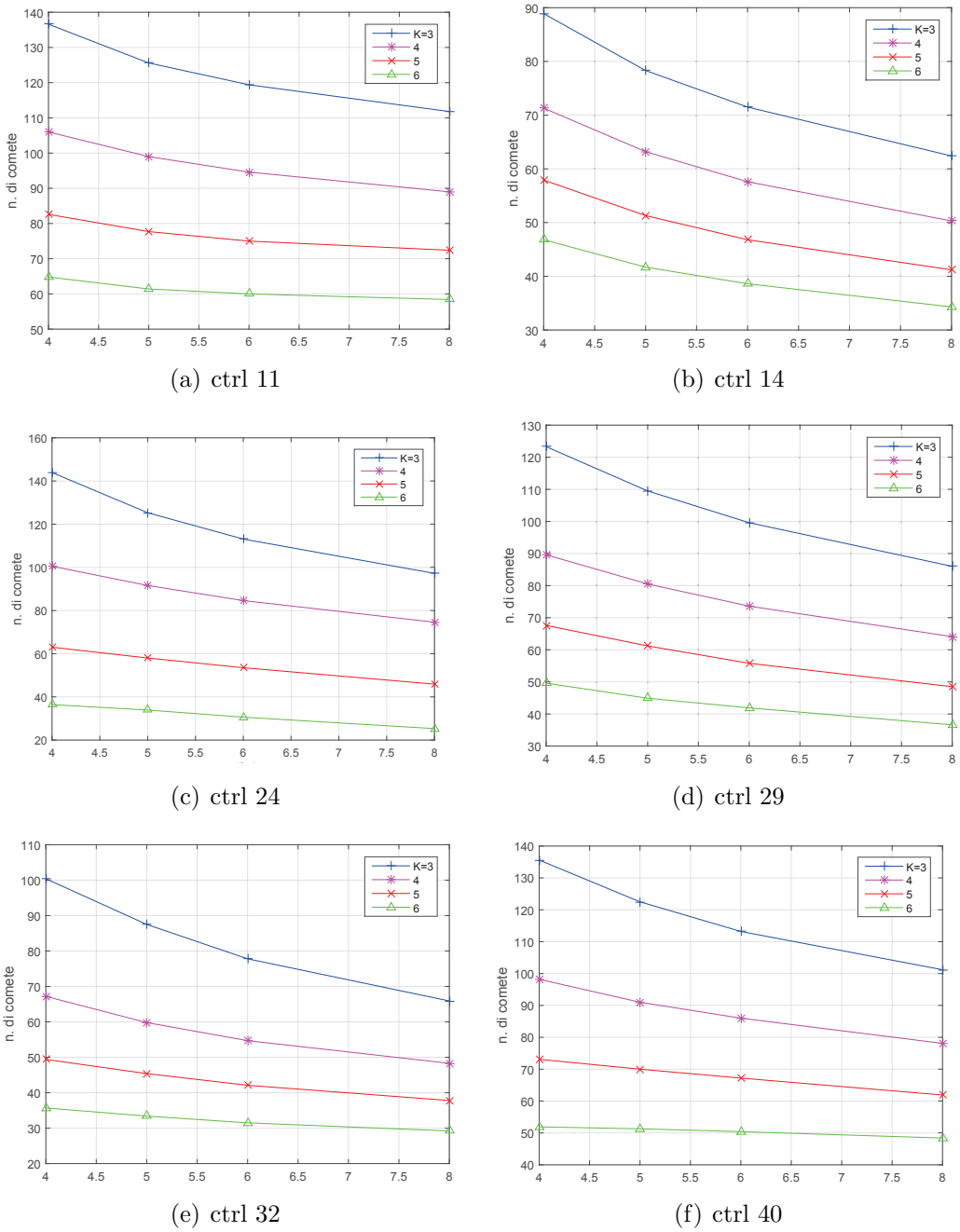
$$D = \frac{M}{1 - 0{,}01 \cdot x}$$

e quindi, misurando quante comete l'algoritmo rileva per una data scelta di parametri, sarà possibile stimare (seppure grezzamente) FP. Per esempio, se si accetta il 10% di falsi positivi e $M = 40$, si dovà avere in media $D = \frac{40}{0.9} = 45$ comete per immagine.

5.1.2 Prove con i dati di controllo

In Fig. 5.1 sono riportati i grafici, ottenuti per tutti i sei *stack* di dati di controllo, del numero medio di comete rilevate in funzione di σ_2 e per diversi valori di K. Si può notare come l'andamento sia decrescente con σ_2, esibendo comunque una tendenza alla saturazione al di sopra di $\sigma_2 = 6$; questo giustifica anche le scelte di *default* riportate in letteratura per tale parametro, che in ogni caso non ha senso far crescere eccessivamente. Molto spiccata risulta invece l'influenza di K, e questo è più che comprensibile in quanto, se si abbassa molto la soglia minima di rilevamento, moltissime macchie spurie, dovute allo sfondo, saranno erroneamente interpretate come comete. D'altra parte, bisogna anche essere cauti nell'elevare eccessivamente K per evitare di incorrere nell'errore opposto, ovvero imporre una soglia minima così elevata che neppure le vere comete siano in grado di superarla.

Alla fine si è deciso di selezionare come valori migliori $\sigma_2 = 5$ e $K = 5$. Con questi due parametri, il numero medio di comete rilevate varia da 45.4 (*stack* "ctrl 40") a 77.7 (*stack* "ctrl 11"). Il rapporto tra il numero di comete rilevate qualora questi due parametri assumano i valori di *default* (σ_2,K) = (4,3) e il numero rilevato con questa scelta dei parametri si aggira intorno a 2 (1.6 in "ctrl 11", 1.5 in "ctrl 14", 2.2 in "ctrl 24", 1.8 in "ctrl 29", 1.9 in "ctrl 32" e 2.7 in "ctrl 40"). Questo significa che con questa scelta si dimezza il numero di comete rilevate, e si ottengono valori medi pari ad alcune decine di comete rilevate, numeri che riteniamo comunque prudenziali, per non incorrere nell'opposto problema di elevare FNR.



(a) ctrl 11

(b) ctrl 14

(c) ctrl 24

(d) ctrl 29

(e) ctrl 32

(f) ctrl 40

Figura 5.1. Numero medio di comete rilevate dei vari *stack* di controllo, in funzione di σ_2

Si è osservato l'andamento del numero di comete rilevate per immagine al variare

dell'indice temporale dello *stack* (Fig. 5.2), e non si è notato un netto comportamento crescente o descrescente, quanto piuttosto un'oscillazione casuale, indicante il fatto che, mentre effettivamente il rumore compromette la fase di rilevamento, il *photobleaching* non lo fa, almeno non in modo evidente. Questo è spiegabile in quanto l'algoritmo funziona sulla base della varianza locale dell'immagine. Certamente, avere un'immagine con minore intensità aumenta le oscillazioni dovute al fatto che c'è più rumore, ma non in modo sistematico.

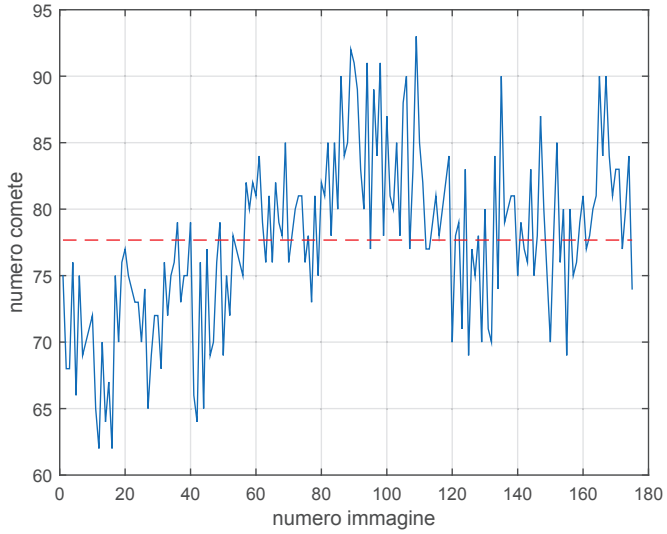


Figura 5.2. Numero di comete rilevate in funzione dell'indice dell'immagine nello *stack*; dati 'ctrl 11'. $\sigma_2 = 5$, $K = 5$. Tratteggiato, il valore medio delle comete rilevate

Notiamo come, ai valori selezionati di σ_2 e K, ci sia comunque un considerevole divario tra il numero massimo e minimo di comete rilevate nelle immagini di uno stesso *stack*. Questo divario è riportato in Tab. 5.1, e rappresenta sicuramente un dato di valenza negativa, in quanto le comete andranno successivamente inseguite all'interno dello *stack*, e il fatto che esse compaiano nelle diverse immagini in numero così variabile complica sicuramente il compito del successivo algoritmo di inseguimento. Come considerazione conclusiva quindi, diciamo che esiste una notevole variabilità di prestazioni dell'algoritmo *watershed* tra un'immagine e l'altra di un singolo *stack* e ovviamente tra *stack* differenti. Questo potrebbe essere dovuto al fatto che lo stadio DoG opera un'insufficiente soppressione di rumore; per questo, più avanti in questo capitolo, si tenteranno altri approcci di soppressione del rumore. La selezione dei parametri σ_2 e K è assai delicata, e comunque le prestazioni sembrano essere fortemente dipendenti dalle immagini considerate. Per questo motivo,

si può azzardare la conclusione che l'algoritmo *watershed* abbia i seguenti vantaggi e svantaggi:

Vantaggi: semplice, computazionalmente efficiente, sintetico (solo due stadi: DoG e *watershed* propriamente detto), si presta a correzioni per esempio modificando lo stadio DoG o migliorando la soppressione di rumore.

Svantaggi: prestazioni fortemente dipendenti dai dati, non stazionarie. Prestazioni fortemente dipendenti dalla scelta dei parametri σ_2 e K.

	ctrl 11	ctrl 14	ctrl 24	ctrl 29	ctrl 32	ctrl 40
Δ_a	42	25	28	27	22	28
Δ_p	35%	40%	38%	36%	39 %	33%

Tabella 5.1. Differenza assoluta Δ_a e percentuale Δ_p tra minimo e massimo numero di comete nei vari *stack*. $\sigma_2 = 5$ e $K = 5$

5.1.3 Prove con i dati di test

Le prove per la selezione dei migliori valori di σ_2 e K sono state eseguite anche per uno degli *stack* di dati di test, ovvero "stack 0". I risultati sono riportati in Tab. 5.2, in termini del minimo e massimo numero di comete rilevate nello *stack*. L'andamento di questi valori è in linea con i risultati ottenuti con i dati di controllo.

	K=3	K=4	K=5	K=6
σ_2=4	130-160	41-75	30-56	24-46
σ_2=5	50-98	35-69	**27-50**	
σ_2=6	42-90	27-62	20-44	
σ_2=8	31-77			

Tabella 5.2. Numero minimo e massimo di comete in *stack 0*. Evidenziati i valori ottenuti con i parametri selezionati $\sigma_2 = 5$ e $K = 5$

A questo punto, sono stati fissati i valori $(\sigma_2,K) = (5,5)$, e l'algoritmo *watershed* è stato applicato a tutti gli *stack* di test, ottenendo i risultati di Tab. 5.3. I valori riportati in questa tabella sono in linea con quelli ottenuti con i dati di controllo. Sussistono le considerazioni già fatte in precedenza, ovvero che il numero di comete rilevato fluttua decisamente all'interno dello *stack*. Il fatto invece che questo numero assuma valori molto diversi tra uno *stack* e l'altro può semplicemente riflettere una diversità intrinseca tra i dati ottenuti in diversi esperimenti.

	stack 0	stack 6	stack 11	stack 14	stack 20	stack 27	stack 30
N. max	50	76	69	69	92	74	71
N. min	27	49	43	39	69	50	48
N. medio	38.0	65.2	55.4	52.9	80.4	62.5	58.6

Tabella 5.3. Numero massimo, minimo, medio di comete rilevate negli *stack* di test con i valori $\sigma_2 = 5$ e $K = 5$

5.2 Algoritmo *Model fitting* gaussiano anisotropo

Questo algoritmo, disponibile all'interno del pacchetto U-track, viene suggerito nel caso in cui l'algoritmo *watershed* dia un numero eccessivo di FP a causa di una forma eccessivamente elongata delle comete. Il parametro principale da fissare è il valore di attendibilità α per i test statistici su cui si basa l'algoritmo. Sono state effettuate prove sia con il valore di *default* di α sia raddoppiandolo per avere maggiore flessibilità (e meno precisione) nei test statistici. In ambo i casi l'algoritmo non rileva praticamente alcuna cometa. Evidentemente il *fitting* con la forma d'onda gaussiana nel nostro caso non sussiste. Le comete non sono elongate, e un eventuale elevato tasso di falsi positivi non va risolto con questo approccio. Probabilmente è assai maggiore l'impatto del rumore granulare, che fa interpretare erroneamente come comete aggregati di pixel dello sfondo (comete che peraltro con ogni probabilità non persistono nelle immagini successive). Inoltre l'algoritmo è anche risultato estremamente impegnativo dal punto di vista computazionale. Di conseguenza, esso non è stato più preso in considerazione.

5.3 Trasformata Anscombe-Wiener

Come spiegato nei Cap. 2 e 4, la trasformata di Anscombe trasforma il processo di Poisson in un processo che si può considerare gaussiano additivo. In questa tesi si è quindi deciso di procedere come segue.

- Calcolare la trasformata di Anscombe di ogni immagine dello *stack*.
- Applicare un filtro di Wiener bidimensionale. Ovviamente questo metodo fornisce una stima ottima del segnale ripulito dal rumore, nell'ipotesi che questo sia AWGN; di conseguenza, questo passaggio, seppure ragionevole, non ha pretese di essere ottimo nel nostro contesto.
- Applicare la trasformata di Anscombe inversa.

Questa procedura è stata applicata a tutti gli *stack* di dati disponibili, sia di test che di controllo, ed è stato calcolato SNR medio sullo *stack* delle immagini così

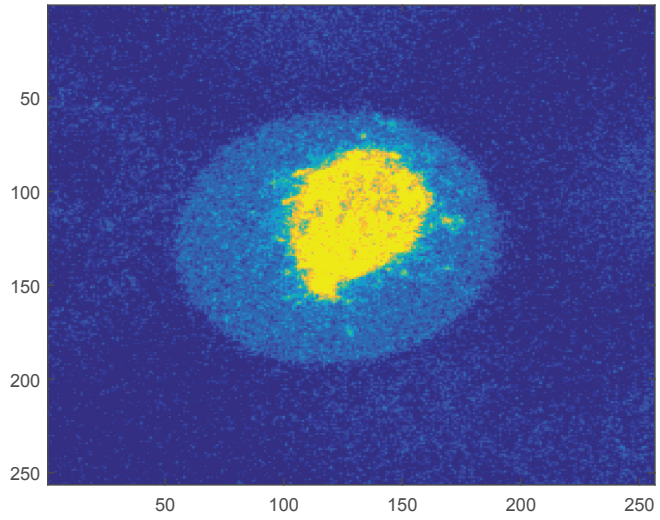
trattate. I risultati sono riportati in Tab. 5.4, insieme ai valori di SNR degli *stack* originari, per consentire un confronto immediato.

Immagini di test							
	stack 0	stack 6	stack 11	stack 14	stack 20	stack 27	stack 30
SNR	16.3	13.1	12.6	13.4	11.8	12.8	13.1
SNR orig	9	6.3	5.7	6.6	4.6	5.8	6.2
guadagno	7.3	6.8	6.9	6.8	7.2	7	6.9
Immagini di controllo							
	ctrl 11	ctrl 14	ctrl 24	ctrl 29	ctrl 32	crtl 40	
SNR	20.8	20.6	10	15.3	14.1	12.7	
SNR orig	14	14	1.8	8.5	7.4	5.7	
guadagno	6.8	6.6	8.2	6.8	6.7	7	

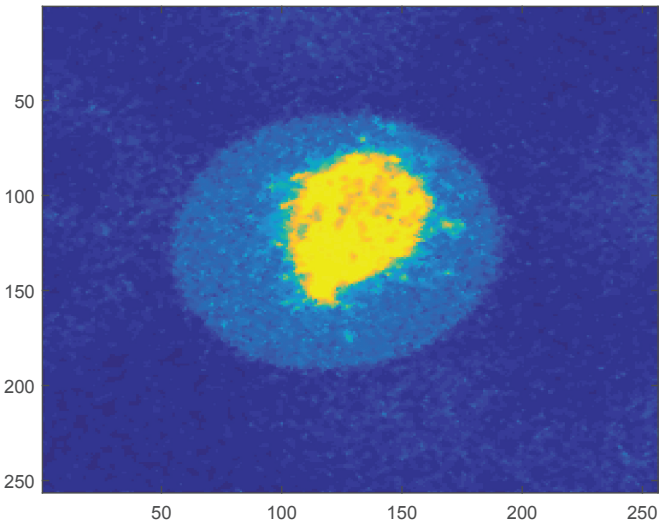
Tabella 5.4. Rapporto segnale-rumore medio dopo pre-elaborazione Anscombe-Wiener (SNR degli *stack* originari sono anche riportati per confronto)

Il guadagno in termini di SNR è vistoso, e rispecchia l'efficacia di questo tipo di algoritmo per l'eliminazione del rumore. Anche qualitativamente, il miglioramento dell'immagine soggetta a questa pre-elaborazione è immediatamente percepibile. In Fig. 5.3 la prima immagine di "stack 0" originale è riportata accanto alla stessa immagine soggetta alla pre-elaborazione. Risultati simili si possono ottenere con qualunque degli *stack* a disposizione, sia di test che di controllo. È bene comunque osservare che, come già accennato, la trasformata di Anscombe inversa introduca una polarizzazione nel valore medio dei dati. Questo effetto, non visualizzabile in Fig. 5.3 dal momento che queste figure sono rappresentate su una scala normalizzata, è comunque piuttosto evidente. A questo problema si può ovviare realizzando una versione modificata della trasformata di Anscombe inversa. In questo lavoro comunque si è deciso di tralasciare questo aspetto, in quanto i successivi stadi di identificazione e inseguimento delle comete lavorano anch'essi su versioni delle immagini normalizzate localmente rispetto alla deviazione standard, e quindi dovrebbero essere relativamente insensibili a questo problema. Similmente, la realizzazione della trasformata di Anscombe modificata, adatta al caso di rumore composito poissoniano e AWGN, è lasciata agli sviluppi futuri del lavoro.

Nonostante il cospicuo guadagno in termini di SNR, è indispensabile validare le prestazioni del metodo in termini di numero di comete rilevate. Infatti, siccome qualunque algoritmo di rilevamento opera in maniera non lineare sui dati, non è scontato il fatto di ritrovare nei risultati variazioni proporzionali alla differenza di SNR riscontrata. Inoltre, l'algoritmo *wateshed* è comunque dotato anch'esso di una intrinseca capacità di ridurre il rumore grazie al filtraggio DoG, e questo può rendere meno vistoso l'effetto di un ulteriore stadio di soppressione del rumore.



(a)



(b)

Figura 5.3. Confronto tra (a) un'immagine di "stack 0" e (b) la corrispondente immagine trattata con trasformata di Anscombe e filtro di Wiener

L'algoritmo di rilevamento *watershed* è stato applicato sui dati pre-elaborati con Anscombe-Wiener, e i risultati in termini di numero medio di comete rilevate sono riassunti in Tab. 5.5. Per completezza, in tabella sono riportati anche il numero medio di comete rilevate dall'algoritmo sui dati non sottoposti alla pre-elaborazione con Anscombe-Wiener, e la differenza percentuale tra i due.

Immagini di test							
	stack 0	stack 6	stack 11	stack 14	stack 20	stack 27	stack 30
n. medio A-W	35.7	62.0	52.1	49.5	73.0	58.5	56.2
n medio orig.	38.0	62.5	55.4	52.9	80.4	62.5	58.6
differenza %	6	0	6	6.4	9.2	6.4	4.1
Immagini di controllo							
	ctrl 11	ctrl 14	ctrl 24	ctrl 29	ctrl 32	crtl 40	
n. medio A-W	60.9	34.2	53.2	57.1	43.4	68.1	
n. medio orig.	77.7	51.2	58.0	61.2	45.4	70.0	
differenza %	21.6	33.2 (*)	8.3	6.7	4.4	2.8	

Tabella 5.5. Numero medio di comete rilevate dall'algoritmo *watershed*. A-W: dati pre-elaborati con Anscombe-Wiener; orig: dati originari. Nota: nel caso (*) si nota un forte rialzo del numero di comete rilevate oltre la metà dello *stack*

Effettivamente un guadagno si consegue (intendendo come "guadagno" una diminuzione del numero di comete rilevate; l'assioma di partenza è comunque che l'algoritmo *watershed* sia affetto da un alto FPR). Tale guadagno si aggira intorno al 6% nella maggior parte dei casi, con alcune eccezioni.

- Lo *stack* di test "stack 6" non mostra di ottenere alcun vantaggio pratico dall'applicazione della pre-elaborazione Anscombe-Wiener.

- Al contrario, nei casi degli *stack* di controllo "ctrl 11" e "ctr14", la pre-elaborazione con Anscombe-Wiener porta a una riduzione molto consistente del numero di comete rilevate. Questi dati di controllo sono effettivamente molto "semplici", e non molto rappresentativi della tipologia dei dati sotto esame. Peraltro, se si studiano dati di controllo anche qualitativamente non troppo dissimili dai dati di test (per esempio, "ctrl 24" o "ctrl 29"), si ottengono valori in linea con quelli ottenuti con i dati di test. Resta la domanda se questa riduzione sia realistica o non significhi invece che si è passati da una condizione di eccesso di FP a una di eccesso di FN, e che quindi alcune comete non siano rilevate. È possibile che, in casi come questi, in cui le immagini sono poco popolate e lo sfondo è facile da distinguere dagli oggetti di interesse, si debba ritoccare il parametro K di soglia minima del *watershed*. Mentre approfondimenti su questa interazione tra ripulitura dal rumore, scelta dei parametri di rilevamento e assestamento di FPR e FNR sono riservati a sviluppi futuri, si coglie l'occasione per sottolineare ancora una volta come non esista di fatto una scelta ottimale dei parametri dell'algoritmo *watershed* (e presumibilmente di qualunque altro algoritmo di rilevamento delle comete),

ma che questa scelta debba essere tarata sulla base della tipologia delle immagini da elaborare. Sicuramente, come anche rimarcato in letteratura, questo è uno degli aspetti più critici da affrontare se si vuole sviluppare un software di analisi automatica che sia veloce, affidabile, robusto e facilmente usabile.

In ogni caso, l'applicazione di una pre-elaborazione con Anscombe-Wiener ha dato risultati preliminari interessanti, e sicuramente merita di essere considerata in analisi più approfondite che prevedano il confronto diretto con una *ground truth*. Questo è lasciato a sviluppi futuri del presente lavoro. Per ora, si può rimarcare come l'aspetto di ripulitura dal rumore sia importantissimo, e come d'altro canto lo stadio DoG che precede il *watershed* vero e proprio provveda comunque di suo a ridurre parzialmente questo rumore.

5.4 Trasformata LOG-Wiener

Come discusso nel Cap. 4, applicare alle immagini un operatore logaritmico trasforma sì il processo di Poisson in rumore additivo, ma senza alcuna evidenza che questo sia né gaussiano né tantomeno bianco. Di conseguenza, applicare in sequenza un operatore come il filtro di Wiener non dà garanzie di ottimalità. Tuttavia, nel seguito si è deciso di applicare anche questo metodo per alcuni motivi. Prima di tutto, mentre sono stati trovati in letteratura risultati, subottimi ma accettabili, di applicazione al caso di rumore *speckle*, non si sono trovati risultati nel caso di rumore di Poisson. Inoltre, questo metodo è semplice e non affetto da problemi di polarizzazione, per cui sembra ragionevole confrontarlo con la trasformata di Anscombe. Infine, dettagli molto fini sulla statistica del rumore che colpisce i nostri dati non sono comunque disponibili, per cui a maggior ragione sembra interessante sperimentare anche questa tecnica. Si è quindi deciso di procedere come segue.

- Applicare il logaritmo in base 10 (scelta arbitraria) a ogni pixel di ogni immagine dello *stack*.
- Applicare un filtro di Wiener bidimensionale all'immagine trasformata.
- Sostituire ad ogni coefficiente $x_{i,j}$ il coefficiente $y_{i,j} = 10^{x_{i,j}}$

Questa procedura è stata applicata a tutti gli *stack* sia di test che di controllo, ed è stato calcolato SNR medio sullo *stack* delle immagini così trattate. I risultati sono riportati in Tab. 5.6, insieme ai valori di SNR degli *stack* elaborati con Anscombe-Wiener e originari per consentire un confronto immediato.

Dall'esame di questa tabella, si può concludere che effettivamente questa tecnica sia poco efficace in termini di miglioramento di SNR, specialmente se confrontata con la trasformata Anscombe-Wiener. Questo conferma le ipotesi sulla natura statistica

Immagini di test							
	stack 0	stack 6	stack 11	stack 14	stack 20	stack 27	stack 30
SNR LOG-W	10.0	6.4	5.8	6.8	4.8	6.0	6.3
SNR A-W	16.3	13.1	12.6	13.4	11.8	12.8	13.1
SNR orig	9	6.3	5.7	6.6	4.6	5.8	6.2
Immagini di controllo							
	ctrl 11	ctrl 14	ctrl 24	ctrl 29	ctrl 32	crtl 40	
SNR LOG-W	14.6	14.2	1.9	8.8	7.6	6.0	
SNR A-W	20.8	20.6	10	15.3	14.1	12.7	
SNR orig	14	14	1.8	8.5	7.4	5.7	

Tabella 5.6. Rapporto segnale-rumore medio dopo pre-elaborazione LOG-Wiener. LOG-W: dati pre-elaborati con LOG-Wiener; A-W: dati pre-elaborati con Anscombe-Wiener; orig: dati originari

dei dati sotto esame, e implicitamente porta alla seguente conclusione: se un processo ha statistica di Poisson, l'applicazione di un operatore logaritmico trasforma il rumore in additivo, ma non in AWGN. Nonostante questi risultati, e in considerazione del fatto che comunque gli algoritmi di rilevamento e inseguimento delle comete non offrono prestazioni legate in modo lineare al rapporto segnale-rumore, si è deciso di provare l'algoritmo *watershed* anche sui dati pre-elaborati con LOG-Wiener. I risultati in termini di numero medio di comete rilevate sono riassunti in Tab. 5.7. Per completezza, in tabella sono riportati anche il numero medio di comete rilevate dall'algoritmo sui dati sottoposti alla pre-elaborazione con Anscombe-Wiener e sui dati originari.

Immagini di test							
	stack 0	stack 6	stack 11	stack 14	stack 20	stack 27	stack 30
n. medio LOG-W	30.1	55.0	46.2	43.6	65.8	51.5	49.1
n. medio A-W	35.7	62.0	52.1	49.5	73.0	58.5	56.2
n medio orig.	38.0	62.5	55.4	52.9	80.4	62.5	58.6
Immagini di controllo							
	ctrl 11	ctrl 14	ctrl 24	ctrl 29	ctrl 32	crtl 40	
n. medio LOG-W	55.8	41.0	51.1	50.1	39.1	61.4	
n. medio A-W	60.9	34.2	53.2	57.1	43.4	68.1	
n. medio orig.	77.7	51.2	58.0	61.2	45.4	70.0	

Tabella 5.7. Numero medio di comete rilevate dall'algoritmo *watershed*. LOG-W: dati pre-elaborati con LOG-Wiener; A-W: dati pre-elaborati con Anscombe-Wiener; orig: dati originari

Si nota come il numero di comete rilevate sia inferiore a quello fornito nel caso di pre-elaborazione con Anscombe-Wiener. Questo conferma una volta di più come la non linearità dell'algoritmo *watershed* renda difficile predirne il comportamento in funzione di SNR, e che quindi, se ci si limita a valutare il numero di comete rilevate partendo dall'ipotesi che il rumore induca un eccessivo FPR, la pre-elaborazone LOG-Wiener dia buoni risultati. Rimane comunque il dubbio di fondo se questi risultati siano affidabili; una loro validazione con confronto con la *ground truth* sembrano inevitabili.

5.5 Deconvoluzione Lucy-Richardson e *blind*

Trattiamo insieme la discussione dei risultati ottenuti con questi due algoritmi in quanto essi si sono rivelati praticamente equivalenti sia dal punto di vista di SNR medio conseguito sui vari *stack*, sia del numero medio di comete identificate dall'algoritmo *watershed* sulle immagini pre-elaborate con deconvoluzione. Tutto sommato ciò non è imprevedibile, in quanto l'algoritmo di deconvoluzione *blind* non fa ipotesi *a priori* sulla PSF (a parte una stima iniziale, dalla quale dipende prevalentemente il tempo di calcolo), mentre l'algoritmo di Lucy-Richardson è ottimizzato per variabili di Poisson. Evidentemente, l'algoritmo *blind* converge a questa stessa soluzione, testimoniando peraltro il fatto che l'ipotesi di statistica di Poisson nel nostro caso sia fondata. In Tab. 5.8 i valori di SNR ottenuti dai due algoritmi di deconvoluzione sono confrontati tra di loro e con quelli degli *stack* originari. Come già accennato, le differenze tra i due metodi di deconvoluzione sono trascurabili. Meno scontato è il fatto che non ci siano praticamente differenze di SNR neppure rispetto ai dati originari. Evidentemente, il contesto in esame è dominato dalla presenza di rumore (come dimostrato dal grosso guadagno ottenuto con la pre-elaborazione Anscombe-Wiener) e non beneficia di un'operazione di deconvoluzione in assenza di opportuna soppressione del rumore.

Anche dal punto di vista qualitativo, non si apprezzano differenze tra le immagini pre-elaborate con ciascuno di questi due metodi di deconvoluzione e le immagini originali (Fig. 5.4).

Per completezza, è stato calcolato il numero medio di comete rilevate dall'algoritmo *watershed* su dati pre-elaborati con ambo i tipi di deconvoluzione. I dati sono riportati in Tab 5.9. È evidente come questa pre-elaborazione non porti differenze significative neppure relativamente al numero di comete rilevate. Nel seguito, questi algoritmi di deconvoluzione non verranno ulteriormente analizzati.

Immagini di test							
	stack 0	stack 6	stack 11	stack 14	stack 20	stack 27	stack 30
SNR R-L	9.1	6.2	5.6	6.4	4.5	5.7	6.1
SNR *blind*	9.1	6.2	5.6	6.4	4.5	5.7	6.1
SNR orig.	9	6.3	5.7	6.6	4.6	5.8	6.2
Immagini di controllo							
	ctrl 11	ctrl 14	ctrl 24	ctrl 29	ctrl 32	crtl 40	
SNR R-L	13.3	13.8	1.7	8.3	7.3	5.6	
SNR *blind*	13.6	13.8	1.7	8.4	7.3	5.6	
SNR orig	14	14	1.8	8.5	7.4	5.7	

Tabella 5.8. SNR dei dati pre-elaborati con la deconvoluzione di Lucy-Richardson (R-L), *blind* e dei dati originari

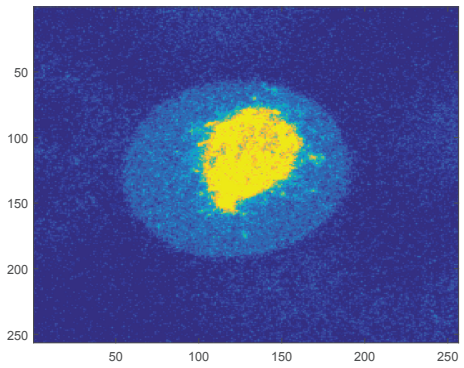
Immagini di test							
	stack 0	stack 6	stack 11	stack 14	stack 20	stack 27	stack 30
n. medio R-L	41	68.5	59.3	56.4	86.4	66.6	61.4
n. medio *blind*	41	68.3	59.2	56.4	86.2	66.4	61.4
n medio orig.	38.0	62.5	55.4	52.9	80.4	62.5	58.6
Immagini di controllo							
	ctrl 11	ctrl 14	ctrl 24	ctrl 29	ctrl 32	crtl 40	
n. medio R-L	82.2	55.2	59.9	60	47.8	72.5	
n- medio *blind*	80.7	55.2	60	60	47.6	72.5	
n. medio orig.	77.7	51.2	58.0	61.2	45.4	70.0	

Tabella 5.9. Numero medio di comete rilevate dall'algoritmo *watershed*. R-L: dati pre-elaborati con deconvoluzione di Lucy-Richardson; *blind*: dati pre-elaborati con deconvoluzione *blind*; orig: dati originari

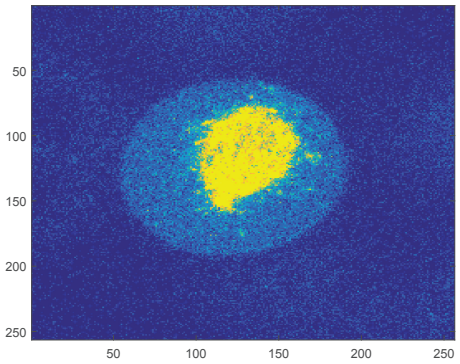
5.6 Algoritmo di inseguimento delle comete

Come già discusso, l'algoritmo selezionato per l'inseguimento delle comete innanzitutto trova la migliore possibile associazione tra particelle presenti in due immagini consecutive, e compone così traiettorie parziali. Queste vengono poi unite a formare traiettorie complete da un secondo stadio, globalmente ottimo [9] [41]. L'algoritmo dipende da numerosi parametri:

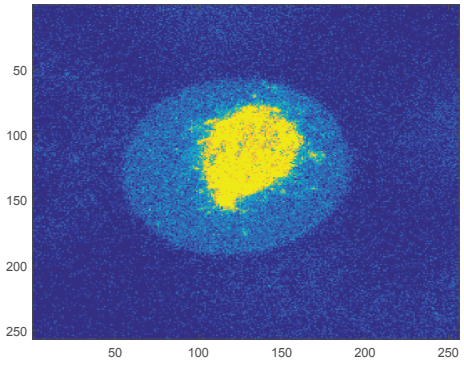
- Δt_{gap}, massima durata dei *gap* tra traiettorie parziali. Questo deve essere abbastanza lungo da permettere di catturare la maggior parte degli eventi di pausa o *shrinkage*, ma abbastanza corto da evitare un'esplosione nella complessità dell'algoritmo dovuta a una proliferazione di possibili passi di legame



(a)



(b)



(c)

Figura 5.4. Confronto tra (a) un'immagine di "stack 0", (b) la corrispondente immagine trattata con deconvoluzione di Lucy-Richardson e (c) trattata con deconvoluzione *blind*

tra traiettorie.

- Minima lunghezza delle traiettorie parziali. Normalmente va bene porlo al valore di 3, per evitare di considerare come traiettorie parziali legami di particelle tra soli due fotogrammi consecutivi, evento caratterizzato da un'elevata probabilità di essere casuale.

- Raggio di ricerca di particelle intorno alla posizione stimata dal filtro di Kalman. Viene calcolata in automatico, è sufficiente imporre i limiti inferiore e superiore, che normalmente è bene non siano troppo stringenti per evitare inutili limitazioni all'algoritmo. Valori tipici: 2-10 pixel.

- Angoli dei due coni di ricerca anteriore e posteriore. In [41] si è evidenziata una scarsa sensibilità dell'algoritmo a questi parametri, che quindi verranno lasciati ai loro valori di *default*: $\pm(25° - 45°)$ e $\pm(10° - 15°)$ (si veda Tab. 3.2).

- È possibile scegliere se l'algoritmo debba considerare traiettorie composite, ovvero l'eventualità che una traiettoria si divida in due (*splitting*), o che due traiettorie si fondano in una sola (*merging*). Siccome questo appesantisce notevolmente l'algoritmo, e anche la notazione in cui vengono espressi i risultati, in questa tesi si è deciso di non abilitare questa opzione.

Di fatto, sia da alcuni test preliminari sia dalle considerazioni riportate in [41], si è potuto concludere come il parametro che condiziona più pesantemente le prestazioni dell'algoritmo sia Δt_{gap}. Sono state eseguite prove con "stack 0" e valori di questo parametro che spaziano da 4 a 15, e, coerentemente con quanto riportato in [41], si è osservato come il numero di traiettorie identificate scenda costantemente, per poi stabilizzarsi per Δt_{gap} intorno a 12. Dando per scontato che l'algoritmo sia affetto da elevato FPR, e che in ogni caso sia preferibile diminuire i falsi positivi anche a costo di subire qualche falso negativo [41], si è selezionato quindi $\Delta t_{gap} = 12$. Peraltro, salendo ulteriormente con questo parametro, al valore di 15 sono stati riscontrati errori nel successivo stadio di riclassificazione delle traiettorie, il che significa che alcune variabili hanno assunto valori non legittimi.

Prima di procedere con la presentazione dei risultati ottenuti, vale la pena di illustrare brevemente le strutture dati generate da U-track per riassumere le informazioni sul rilevamento e sull'inseguimento, e il tipo di indagini statistiche svolte.

5.6.1 Strutture dati di U-track

La prima struttura dati importante è quella che contiene l'uscita dello stadio di rilevamento delle comete. Questa struttura MATLAB, denominata **movieInfo**, consta

di tante righe quante sono le immagini (normalmente 120) e di tre campi principali: **xCoord**, **yCoord**, **amp**. Questi campi contengono rispettivamente le stime delle coordinate spaziali e dell'ampiezza (e le relative deviazioni standard) delle n_i particelle trovate nel fotogramma i-esimo preso in considerazione. In una elaborazione esemplificativa di "stack 0", il campo **movieInfo(1).xCoord** è composto da 34 righe e 2 colonne, le quali contengono la coordinata X e rispettiva deviazione standard delle 34 particelle trovate nel fotogramma 1.

La seconda struttura dati è quella fornita dallo stadio di inseguimento vero e proprio, ed è denominata **tracksFinal**. Si tratta di una struttura MATLAB contenente una riga per ciascuna delle traiettorie identificate (per esempio, nel caso di "stack 0", 411 righe)[1] e tre campi.

1. Il primo campo, **tracksFearIndxCG**, è una matrice di connettività, ovvero un vettore con tanti elementi quante sono le immagini connesse a formare la traiettoria (ovvero, pari alla lunghezza della traiettoria). Ogni elemento rappresenta il numero della particella coinvolta nel congiungimento tra i due fotogrammi (il primo e il secondo, il secondo e il terzo ecc., a seconda della posizione dell'elemento nel vettore). Per esempio, se si elabora "stack 0", si ottiene **tracksFeatIndxCG(1)** = [10,7,4,4,6,5,4,5,6]. Questo significa che la prima traiettoria trovata è lunga 9 immagini, e connette la particella 10 dell'immagine i (le cui coordinate **xCoord**, **yCoord** e ampiezza **amp** sono reperibili nella decima riga di **movieInfo(i)**) con la particella 7 dell'immagine $i + 1$ e via di questo passo.

2. Il secondo campo, **tracksCoordAmpCG**, fornisce coordinate X e Y, ampiezza e rispettive deviazioni standard di ciascuna delle particelle coinvolte nella traiettoria. In pratica, esso riproduce in forma ordinata l'informazione che sarebbe comunque reperibile in **movieInfo(i)**).

3. Il terzo campo è un vettore di cui si riporta un esempio:

   $$[1,1,1,NaN,9,2,1,NaN]$$

   Esso dà i dettagli della traiettoria, ovvero:

   - La traiettoria inizia dall'immagine 1.
   - Questa immagine rappresenta l'inizio della traiettoria.
   - L'indice della traccia coinvolta è 1 (può essere diverso da uno solo se sono ammesse traiettorie composite).

[1] Come accennato, in queste prove non si considera la possibilità di avere traiettorie composite, che derivano da uno *splitting* o *merging* di altre traiettorie. Se questa opzione fosse consentita, la struttura dati **tracksFinal** risulterebbe più complessa.

- Si tratta di un vero inizio, se il campo è *NaN - not a number*. In caso di traiettorie composite potrebbe trattarsi di un evento di *splitting*, e questo campo riporterebbe il numero, tra quelle possibili, della traiettoria che entra a fare parte della traiettoria composita.
- La traiettoria termina all'immagine 9.
- L'indice della traccia coinvolta è 1 (può essere diverso da uno solo se sono ammesse traiettorie composite).
- Si tratta di un vero termine di traiettoria, se il campo è *NaN - not a number*. In caso di traiettorie composite potrebbe trattarsi di un evento di *merging*, e questo campo riporterebbe il numero, tra quelle possibili, della traiettoria che entra a fare parte della traiettoria composita.

La terza struttura riassume tutta una serie di dati statistici ottenuti dopo un'operazione di post-elaborazione volta a riclassificare le traiettorie sulla base della natura dei *gap*. Infatti, un *gap* può verificarsi anche per ragioni diverse da una pausa o una fase di *shrinkage*, per cui si ritiene necessaria una riclassificazione di questi *gap*.

La velocità della particella nell'intervallo temporale corrispondente al *gap*, ipotizzata costante, viene stimata per interpolazione dai dati di velocità nelle immagini subito adiacenti. Quindi, le velocità istantanee per ogni traccia sono determinate calcolando lo spostamento tra immagini consecutive. Per la riclassificazione, si valuta anzitutto la distribuzione della velocità stimata durante i *gap*. Spostamenti dell'estremità *plus* del MT che portano a una velocità superiore al 70% della velocità che si aveva appena prima della scomparsa della particella denotano presumibilmente una continuazione della crescita, per cui si riclassifica *fgap* come fase di crescita; altrimenti, si conferma la classificazione come *fgap*. Normalmente queste situazioni sono dovute a un mancato rilevamento della particella, o al fatto che la medesima si sia temporaneamente spostata fuori dal piano focale. Similmente si agisce per riclassificare i *bgap*. Se lo spostamento dell'estremità *plus* del MT porta a un valore di velocità minore del 95° percentile della distribuzione delle velocità dei *bgap* classificati come veri eventi di *shrinkage*, allora si ritiene di trovarsi di fronte a un evento di pausa e non di *shrinkage*, e come tale il *gap* viene riclassificato.

Dopo la riclassificazione, i profili delle traiettorie sono riportati in una matrice le cui colonne contengono: l'indice della traiettoria, il tipo (crescita, *fgap*, ecc.), le immagini iniziali e finali, le velocità medie in μm/min, la durata media in s, lo spostamento totale (lunghezza della traiettoria) in μm. Tutte queste informazioni sono riportate nella variabile **projData**, insieme a un riassunto dei parametri impostati, al numero totale di traiettorie, all'intensità e all'area delle comete congiunte (**featArea, featInt**), alle percentuali di *fgap* e *bgap*, alle percentuali di riclassificazione ecc. I dati statistici forniti da *projData*, che di fatto è la più importante variabile di uscita di U-track, sono riportati in Appendice A per tutte le principali analisi svolte

in questa tesi. Il significato della maggior parte delle statistiche è auto-esplicante. Alcuni dati statistici sono riassunti in istogrammi del tipo di quelli rappresentati in Fig. 5.5.

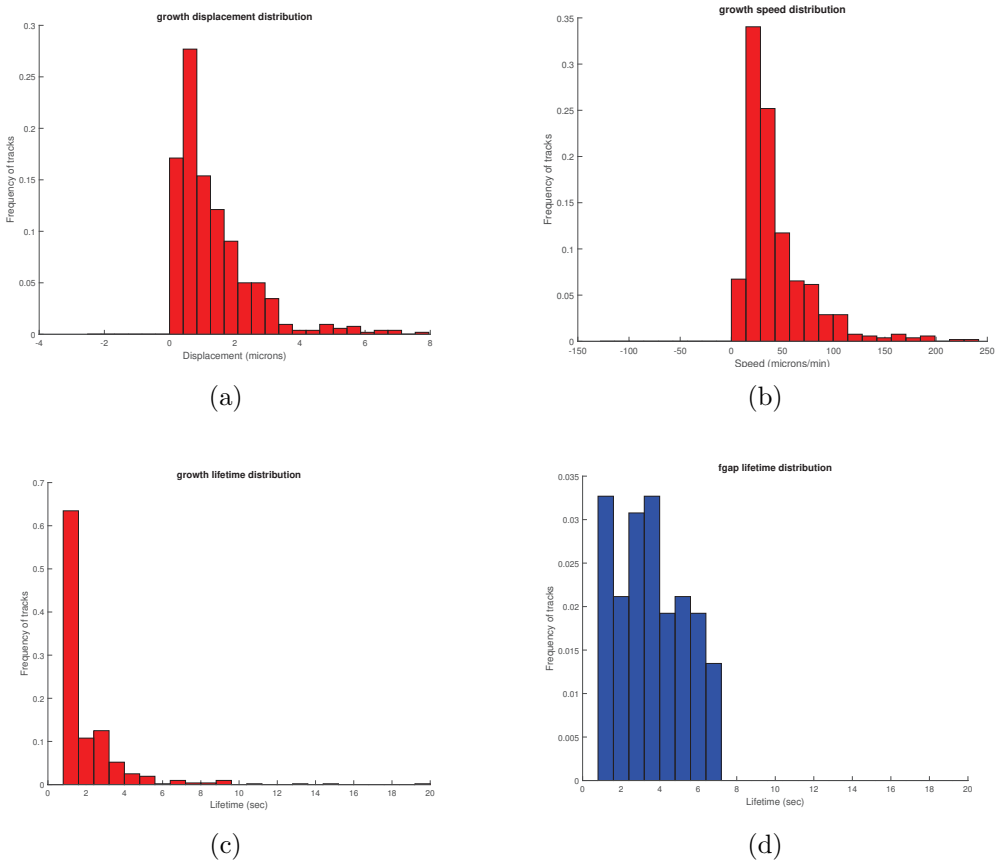


Figura 5.5. Fase di crescita, "stack 0". Distribuzione di spostamento (a), velocità (b), durata (c). Fase *fgap*, distribuzione di durata (d)

Oltre alle strutture dati descritte, U-track mette a disposizione strumenti grafici per visualizzare la sequenza delle immagini, a cui si possono sovrapporre i risultati conseguiti dall'algoritmo: le comete, le tracce identificate, la cumulativa di tutte tali tracce, le tracce riclassificate. Sono possibili diverse scelte di simboli e colori per evidenziare: comete, inizio e fine delle tracce, *fgap*, *bgap*, e altri fenomeni di secondo livello, come ad esempio se i *gap* sono più corti o più lunghi delle traiettorie parziali che li precedono o seguono (il primo caso considerato buono, il secondo indice di minore affidabilità), oppure se una cometa è presente in tutte le immagini (bianca), oppure a un certo punto compare e poi scompare (rossa), se compare non nella prima immagine ma poi persiste (gialla), se compare nella prima immagine ma

poi scompare (viola). Le varie informazioni possono essere sovrapposte nella stessa sequenza di immagini oppure rappresentate separatamente. Un esempio è riportato in Fig. 5.6.

5.6.2 Selezione dei parametri e validazione dell'algoritmo

Un primo esempio di valutazione delle traiettorie è stato effettuato con "stack 0" e i seguenti parametri:

- Per il rilevamento: $\sigma_2 = 5$, $K = 5$.
- Per l'inseguimento: $\Delta t_{gap} = 12$, $MLTS = 3$ (quest'ultimo rappresenta la minima lunghezza affinché una traiettoria parziale venga considerata tale).

Sono risultate numerose traiettorie, con ben 520 fasi di crescita inseguite dall'algoritmo. In particolare, data la dimensione del pixel (circa 160 nm), una traiettoria risulta avere in media una lunghezza pari a circa 8 pixel. Per evitare una proliferazione di traiettorie corte, si è provato a porre $MLTS = 4$; in questo modo, traiettorie parziali che si estendono in 1 - 2 oppure 3 immagini consecutive vengono considerate spurie. Da un'indagine preliminare, effettuata sempre con "stack 0", si è visto come questa strategia sia efficace per ridurre il numero di traiettorie inseguite: anche visivamente si apprezza una riduzione netta di traiettorie sospette di essere falsi positivi. La velocità media diminuisce lievemente, mentre aumentano sia la lunghezza che la durata media. Sono quindi stati selezionati questi parametri definitivi:

- $\sigma_2 = 5$, $K = 5$.
- $\Delta t_{gap} = 12$, $MLTS = 4$.
- Disabilitata l'opzione *merge/split* (che si è rivelata ininfluente nelle prove preliminari).
- Tutti i restanti parametri sono stati lasciati ai loro valori di *default*.

I risultati ottenuti sono riepilogati nelle Tab. 5.10 e 5.11 per quanto riguarda rispettivamente i dati di test e di controllo , sia originali che pre-elaborati con Anscombe-Wiener e LOG-Wiener.

I valori medi ottenuti sugli *stack* di test e di controllo sono stati confrontati con valori medi ottenuti manualmente dal laboratorio (Tab. 5.12). Sui dati di test, non pre-elaborati per evidenziare solo l'effetto dello stadio di inseguimento, si ottiene:

Velocità: Valor medio 0.71 μm/s, deviazione standard (STD) 0.47 μm/s. Intervallo $\pm\sigma$: 0.24 - 1.18 μm/s.

	Tracce	Velocità	V-STD	Durata	D-STD	lunghezza	L-STD
stack 0 orig	277	44.6	29.1	2.5	1.5	1.8	1.3
stack 0 A-W	264	43.4	26.9	2.6	1.5	1.7	1.2
stack 0 LOG-W	217	41.0	26.6	2.6	1.4	1.7	1.2
stack 6 orig	520	33.2	23.6	2.5	1.8	1.3	1.1
stack 6 A-W	494	32.5	20.4	2.6	1.8	1.4	1.1
stack 6 LOG-W	442	32.4	22.6	2.5	1.6	1.3	1.0
stack 11 orig	383	55.5	43.0	2.6	2.1	2.1	1.7
stack 11 A-W	356	53.7	43.9	2.6	1.6	2.1	1.7
stack 11 LOG-W	313	52.5	43.1	2.7	1.7	2.1	1.6
stack 14 orig	375	41.4	27.8	2.6	1.5	1.7	1.3
stack 14 A-W	344	39.4	28.3	2.7	1.8	1.6	1.3
stack 14 LOG-W	327	37.6	22.7	2.5	1.4	1.5	1.0
stack 20 orig	590	41.4	28.5	2.5	1.4	1.6	1.2
stack 20 A-W	553	40.7	30.0	2.6	1.7	1.6	1.3
stack 20 LOG-W	495	40.0	32.9	2.5	1.4	1.6	1.2
stack 27 orig	430	39.9	26.2	2.5	1.5	1.6	1.2
stack 27 A-W	423	39.2	25.5	2.5	1.4	1.5	1.1
stack 27 LOG-W	376	37.5	25.9	2.5	1.5	1.5	1.0
stack 30 orig	399	44.0	28.5	2.3	1.2	1.6	1.2
stack 30 A-W	371	42.6	27.5	2.4	1.3	1.6	1.2
stack 30 LOG-W	345	43.3	26.3	2.4	1.6	1.7	1.2

Tabella 5.10. Numero tracce rilevate, velocità [μm/min], durata [s], lunghezza [μm] e rispettive deviazioni standard. *Stack* di test originali, pre-elaborati con Anscombe-Wiener (A-W), pre-elaborati con LOG-Wiener (LOG-W)

Durata: Valor medio 2.5 s, STD 1.6 s. Intervallo $\pm\sigma$: 0.9 - 4.1 s.

Lunghezza: Valor medio 1.7 μm, STD 1.3 μm. Intervallo $\pm\sigma$: 0.4 - 3 μm.

Risultati analoghi (riportati in Tab. 5.12) si ottengono elaborando i dati di controllo. Si possono fare le seguenti osservazioni.

- L'algoritmo trova valori medi di velocità superiori a quelli del Laboratorio, mentre i valori di durata media sono inferiori e le lunghezze confrontabili.

- Ancora più interessante, le deviazioni standard dei dati forniti dall'algoritmo, per tutte le misure, sono maggiori di quelle ottenute dalle misure del Laboratorio. Infatti, il rapporto tra valore medio e deviazione standard si aggira

	Tracce	Velocità	V-STD	Durata	D-STD	lunghezza	L-STD
ctrl 11 orig	861	36.8	32.1	2.7	1.8	1.5	1.3
ctrl 11 A-W	464	30.5	28.0	2.5	1.5	1.2	1.0
ctrl 11 LOG-W	521	29.8	23.0	2.7	1.8	1.2	0.9
ctrl 14 orig	379	38.8	31.7	2.7	1.9	1.6	1.3
ctrl 14 A-W	255	34.5	29.1	2.7	1.9	1.5	1.3
ctrl 14 LOG-W	319	29.7	22.1	2.7	1.8	1.3	1.0
ctrl 24 orig	473	29.6	22.2	2.6	1.6	1.2	1.0
ctrl 24 A-W	418	29.0	20.1	2.8	1.8	1.3	1.1
ctrl 24 LOG-W	425	30.5	22.0	2.8	1.9	1.4	1.2
ctrl 29 orig	485	37.4	25.4	2.4	1.3	1.5	1.2
ctrl 29 A-W	424	35.5	24.6	2.5	1.4	1.4	1.0
ctrl 29 LOG-W	408	34.5	23.8	2.4	1.1	1.3	0.9
ctrl 32 orig	328	37.4	24.2	2.5	1.9	1.5	1.1
ctrl 32 A-W	317	36.9	23.4	2.6	2.1	1.5	1.3
ctrl 32 LOG-W	292	35.7	21.1	2.6	1.9	1.4	1.0
ctrl 40 orig	542	37.6	26.5	2.5	1.4	1.5	1.2
ctrl 40 A-W	539	35.8	23.9	2.5	1.3	1.4	1.1
ctrl 40 LOG-W	479	36.7	22.0	2.5	1.4	1.5	1.0

Tabella 5.11. Numero tracce rilevate, velocità [μm/min], durata [s], lunghezza [μm] e rispettive deviazioni standard. *Stack* di controllo originali, pre-elaborati con Anscombe-Wiener (A-W), pre-elaborati con LOG-Wiener (LOG-W)

intorno al 30-40% sui dati del Laboratorio, e intorno al 60-70% sui risultati dell'algoritmo.

- Quindi, l'algoritmo fornisce dati con maggiore dispersione, velocità mediamente maggiori e durate inferiori. Evidentemente, esso capta anche traiettorie veloci e sfuggenti che possono non essere visualizzate da un osservatore.

Bisogna comunque notare che altre misurazioni effettuate sempre su dati di controllo e su dati di test trattati con CITK siRNA marcati con EB3-dtTomato, analoghi a quelli degli esperimenti che hanno fornito gli *stack* elaborati in questa tesi, hanno portato a valori di velocità media decisamente superiori: 0.7 μm/s per i dati di controllo, e addirittura 1.4 μm/s per i dati di test (dati non ancora pubblicati). Esperimenti simili ma con MT marcati con EB1-GFP hanno fornito valori di velocità ancora differenti [43]. Quindi, è possibile che in realtà i valori di velocità media ottenuti dall'algoritmo non siano irrealistici, ma piuttosto rispecchino una potenziale eterogeneità, e forse anche una bassa affidabilità dei valori elaborati manualmente, che risultano presumibilmente troppo bassi.

Comunque, assumendo per un momento che i dati elaborati manualmente e riportati in Tab. 5.12 siano corretti, ci si può chiedere se la discrepanza possa essere dovuta al fatto che la seconda parte dello *stack* non è stata usata nelle valutazioni manuali in quanto considerata eccessivamente rumorosa e affetta da *photobleaching*. Quindi, sono state rifatte le misurazioni su "stack 0" usando solamente le prime 60 immagini dello *stack*. I risultati sono i seguenti:

Velocità: Valor medio 0.58μm/s, STD 0.46 μm/s. Intervallo $\pm\sigma$: 0.12 - 1.04 μm/s.

Durata: Valor medio 2.4 s, STD 1.4 s. Intervallo $\pm\sigma$: 1.0 - 3.8 s.

Lunghezza: Valor medio 1.4 μm, STD 1.0 μm. Intervallo $\pm\sigma$: 0.4 - 2.4 μm.

Il valor medio della velocità diminuisce lievemente, ma la deviazione standard rimane praticamente uguale (quindi, percentualmente pesa ancora di più). La durata non viene alterata in modo significativo; di conseguenza, la lunghezza diminuisce sia in media che come deviazione standard. Questo sembra indicare un ruolo del *photobleaching* nel rendere instabili questo tipo di misure, anche se l'interpretazione precisa di questi dati sarà discussa in dettaglio con il personale del Laboratorio, ed è demandata a sviluppi futuri di questo lavoro. Le velocità medie calcolate sui dati elaborati con Anscombe-Wiener e LOG-Wiener sono lievemente inferiori, ma confrontabili con quelle ottenute sui i dati originari: 0.69 μm/s con STD 0.47 μm/s nel caso di dati Anscombe-Wiener, e 0.68 μm/s con STD 0.48 μm/s nel caso LOG-Wiener. Anche le altre metriche non cambiano in misura sostanziale.

	Velocità	V-STD	Durata	D-STD	Lunghezza	L-STD
Test Lab	0.40	0.14	5.7	2.6	2.1	0.7
Ctrl Lab	0.41	0.20	6.3	3.5	2.2	0.9
Test Algo	0.71	0.47	2.5	1.6	1.7	1.3
Ctrl Algo	0.55	0.37	2.6	1.7	1.4	1.2

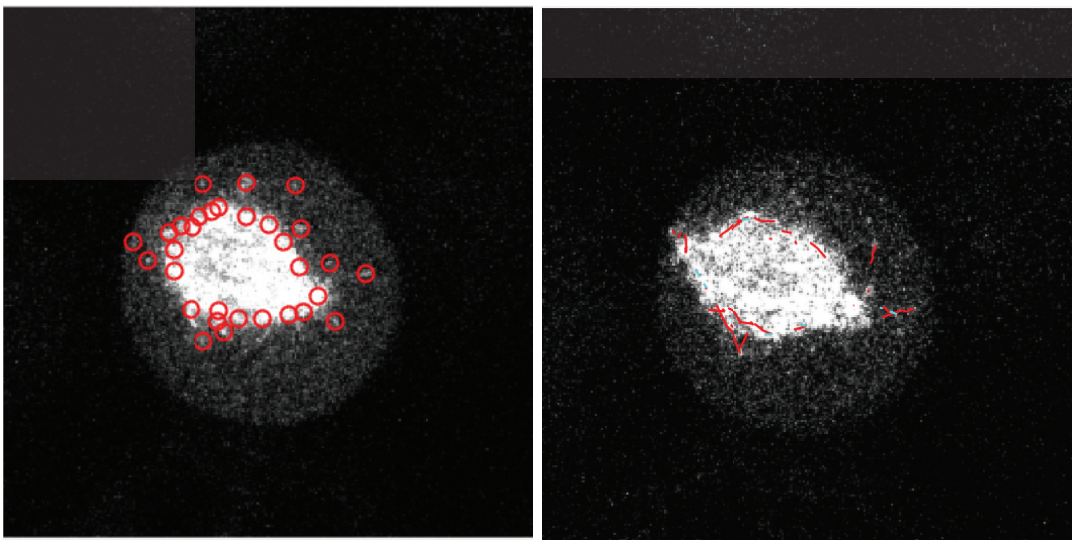
Tabella 5.12. Velocità [μm/s], durata [s], lunghezza [μm] e rispettive deviazioni standard. Dati di riferimento del Laboratorio e valori medi ottenuti dagli algoritmi descritti

Nell'elaborare i risultati, sono anche state visionate le sequenze a cui sono state sovrapposte le traiettorie. Si è notato come, nella zona di interesse (MT astrali), le traiettorie inseguite non siano numerose, a dispetto del numero totale elevato. Per esempio, in "stack 20" si rilevano poche traiettorie utili, e effettivamente questo *stack* è caratterizzato da bassissimo SNR. Anche gli *stack* di controllo "ctrl 11" e "ctrl 14", che esibiscono caratteristiche macroscopiche diverse dai dati di test, presentano questo comportamento.

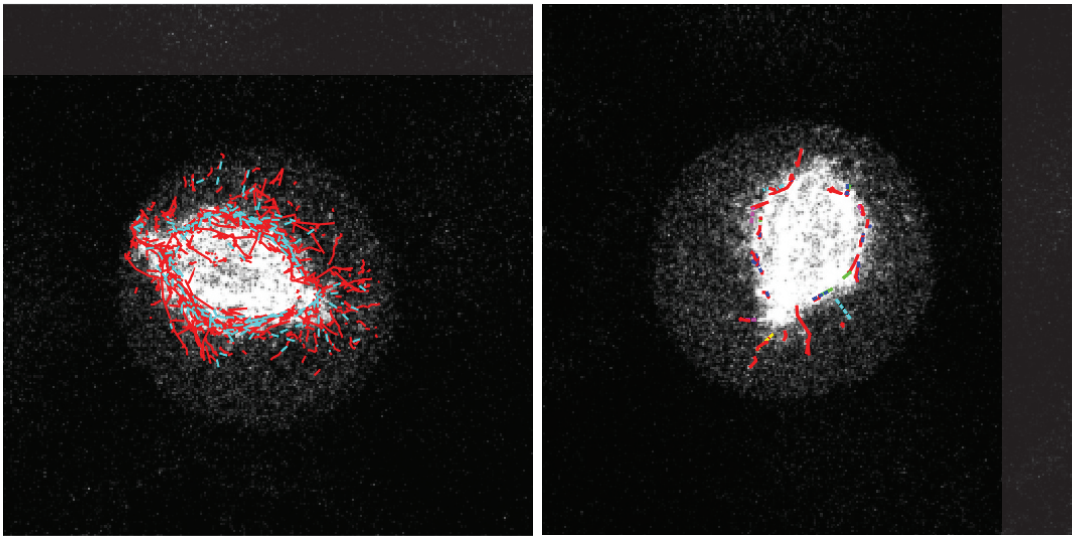
5.7 Commenti

Possiamo fare le seguenti considerazioni finali.

- Il numero di traiettorie rilevate non è legato molto strettamente a quella che si percepisce soggettivamente essere la qualità del rilevamento.
- Almeno alcuni dei parametri vanno stabiliti con estrema cura a seconda della tipologia delle immagini. Un'analisi più accurata di questo aspetto (e su come si possa classificare *a priori* un'immagine per poter orientare la selezione dei parametri) è lasciata agli sviluppi futuri di questo lavoro.
- Invece, sembra bene limitare il numero delle comete rilevate, in quanto si è notato che questo numero, se eccessivo, complica notevolmente l'inseguimento delle traiettorie.
- Le traiettorie seguite dall'algoritmo mediamente sono più veloci e di più breve durata rispetto a quelle misurate manualmente sui dati di test.
- Il *photobleaching* ha certamente un effetto sulle prestazioni dell'algoritmo, dal momento che elaborando solo le prime 60 immagini dello *stack* i valori di velocità media diminuiscono lievemente. Questo è vero anche se metriche come SNR di fatto non sono in grado di misurare questo tipo di degradazione lungo l'asse temporale dello *stack*.
- Le immagini pre-elaborate con Anscombe-Wiener o LOG-Wiener forniscono valori di velocità media confrontabili con quelli dei dati originari. Di conseguenza, anche l'aspetto relativo all'influenza del rumore su queste misure merita ulteriore approfondimento.
- Infine, bisogna rimarcare che il fatto che l'algoritmo insegua e conteggi traiettorie brevi e veloci (e che quindi le traiettorie siano caratterizzate da grande variabilità, rispecchiata dagli elevati valori delle STD) potrebbe non essere "un errore", ma piuttosto riflettere una situazione obiettiva, che però le misurazioni manuali non sono in grado di rilevare.



(a) (b)



(c) (d)

Figura 5.6. Esempi di immagini fornite da MATLAB con informazioni relative alle comete (a), alle traiettorie (b) alle traiettorie statiche (c) e alle traiettorie riclassificate (d)

Capitolo 6

Conclusioni

6.1 Discussione dei risultati

Questo lavoro di tesi ha affrontato il problema della identificazione automatica dei microtubuli astrali in sequenze di immagini di microscopia a fluorescenza. Si tratta di un problema molto interessante, in quanto attualmente l'analisi di queste immagini viene effettuata in modo manuale, e questo lavoro, oltre ad essere tedioso, stancante e poco riproducibile, sottrae energie preziose ad attività più importanti. Di fatto, l'analisi dei dati sta diventando il collo di bottiglia dell'intero processo di ricerca.

Peraltro, al momento attuale non esistono algoritmi in grado di analizzare questo tipo di immagini con sufficiente affidabilità e duttilità. Al contrario, esistono ottimi algoritmi che si adattano benissimo alla soluzione di specifici problemi, ma le cui prestazioni si riducono sensibilmente se cambia (anche non di molto) la tipologia dei dati. Altri metodi sono più generali, ma richiedono comunque che una serie di parametri critici vengano tarati con cura, pena un drastico crollo delle prestazioni. Per di più, anche la valutazione precisa delle prestazioni di questi algoritmi non è semplice, in quando spesso mancano i riferimenti (*ground truth*) e manca accordo su quali metriche siano più rappresentative.

In questo lavoro:

- È stata svolta un'accurata analisi della numerosa letteratura sull'argomento. Sono stati descritti gli algoritmi più significativi, ed è stata tentata una loro classificazione sulla base dei principali aspetti del problema che essi si propongono di risolvere: limitazione del rumore, miglioramento del segnale, tecnica di rilevamento vero e proprio, tecnica di inseguimento.

- È stata eseguita una ricerca di software disponibile, ed è stato selezionato il *toolbox open source* di MATLAB denominato U-track, sviluppato dal *Department of Cell Biology* della *Harvard Medical School*.

- Sono stati selezionati l'algoritmo *watershed* per l'identificazione delle comete e l'algoritmo di inseguimento a due stadi realizzato in U-track. Sono state realizzate alcune funzioni MATLAB aggiuntive per migliorare la qualità del rilevamento: trasformata Anscombe-Wiener, trasformata LOG-Wiener, decorrelazione Lucy-Richardson, decorrelazione cieca.

- Sono state svolte intensive campagne di misura sui sette *stack* di test e sui sei *stack* di controllo forniti dal Laboratorio. Per tali esperimenti sono state adottate metriche basate su valori attesi dei parametri da valutare (numero di comete rilevate, velocità, durata e lunghezza delle tracce).

6.2 Considerazioni conclusive e sviluppi futuri

I risultati ottenuti hanno evidenziato in particolare alcuni aspetti del problema, e si sono dimostrati piuttosto in linea con le considerazioni riportate in letteratura.

- Al momento attuale, non si è ancora nelle condizioni di poter adottare un algoritmo di rilevamento automatico senza la necessità di validarlo con osservazione diretta, dal momento che le prestazioni di tutti gli algoritmi esistenti (inclusi quelli analizzati in questa tesi) sono fortemente dipendenti dalle caratteristiche dei dati.

- Analogamente, per ora non è praticabile un approccio che non consideri separatamente la fase di rilevamento degli oggetti nelle singole immagini, seguita da un algoritmo che ricostruisca le traiettorie di tali oggetti. Anche se ci sono evidenze che questo approccio non sia quello quello spontaneamente adottato dall'osservatore umano, per il momento non è pensabile affrontare in un unico passo la complessità del problema.

- È nostra convinzione che sia indispensabile eseguire un'attenta soppressione del rumore prima di elaborare le immagini. Dalle prove eseguite in questa tesi, si sono osservati grossi miglioramenti a seguito di una maggior attenzione posta su questo aspetto di pre-elaborazione, specie relativamente alla fase di rilevamento degli oggetti nelle singole immagini.

- Per quanto riguarda la fase di inseguimento delle traiettorie, si sono ottenuti valori di velocità media superiori e di durata media inferiori a quelli valutati manualmente. Tuttavia, altre misurazioni manuali su dati simili (colture trattate con CITK siRNA e marcate con EB3-dtomato) hanno fornito velocità medie superiori, e questo induce a ritenere possibile che l'algoritmo abbia trovato valori medi realistici. Anche le deviazioni standard di questi parametri

si sono rivelate più grandi, il che significa che, in ogni caso, l'algoritmo individua traiettorie molto eterogenee tra loro. Il *photobleaching* ha sicuramente un ruolo importante in questo fatto, il rumore invece sembra meno rilevante.

- Rimane comunque discutibile se il fatto di identificare traiettorie così numerose ed eterogenee sia da considerarsi un limite dell'algoritmo, o piuttosto dell'analisi manuale, che non è in grado di seguire fenomeni così veloci. È quindi estremamente importante affrontare il discorso della validazione degli algoritmi su dati reali provvisti di affidabile *ground truth*. Questo aspetto, molto discusso anche nella letteratura recente, mette in evidenza la necessità di impostare in questi termini anche i confronti tra i vari algoritmi proposti, confronti attualmente molto difficili da effettuare proprio per l'eterogeneità sia dei dati di prova che delle metriche usate per la validazione.

Appendice A

Statistiche del rilevamento

A.1 Statistiche di "Stack 0"

Dati originali

```
medNNdistWithinFrameMic 1.2122
nGrowths 585
growth_speed_median 46.4462
growth_speed_mean 61.9983
growth_speed_std 49.1174
growth_lifetime_median 2
growth_lifetime_mean 2.88034
growth_lifetime_std 2.11126
growth_length_median 2.02726
growth_length_mean 2.68285
growth_length_std 2.3217
nFgaps 83
fgap_speed_median -3.95946
fgap_speed_mean -1.92052
fgap_speed_std 26.6691
fgap_lifetime_median 4
fgap_lifetime_mean 3.91566
fgap_lifetime_std 1.75794
fgap_length_median -0.2743
fgap_length_mean 0.0733969
fgap_length_std 1.51671
GrowthSpeedBeforeFgap_MicPerMin_mean 58.8477
GrowthSpeedBeforeFgap_MicPerMin_SE 3.86957
GrowthLifetimeBeforeFgap_Sec_mean 2.94643
GrowthLifetimeBeforeFgap_Sec_SE 0.239528
GrowthLengthBeforeFgap_Mic_mean 2.63196
GrowthLengthBeforeFgap_Mic_SE 0.213131
fgap_freq_time_mean 0.462207
fgap_freq_time_SE 0.0213284
```

```
fgap_freq_length_mean 0.68603
fgap_freq_length_SE 0.0759325
nBgaps 0
bgap_speed_median NaN
bgap_speed_mean NaN
bgap_speed_std NaN
bgap_lifetime_median NaN
bgap_lifetime_mean NaN
bgap_lifetime_std NaN
bgap_length_median NaN
bgap_length_mean NaN
bgap_length_std NaN
bgap_freq_time_mean NaN
bgap_freq_length_mean NaN
bgap_freq_time_SE NaN
bgap_freq_length_SE NaN
GrowthSpeedBeforeBgap_MicPerMin_mean NaN
GrowthSpeedBeforeBgap_MicPerMin_SE NaN
GrowthLifetimeBeforeBgap_Sec_mean NaN
GrowthLifetimeBeforeBgap_Sec_SE NaN
GrowthLengthBeforeBgap_Mic_mean NaN
GrowthLengthBeforeBgap_Mic_SE NaN
nGrowthTermEvents 476
GrowthSpeedBeforeTermEvent_MicPerMin_mean 63.1336
GrowthSpeedBeforeTermEvent_MicPerMin_SE 2.37501
GrowthLifetimeBeforeTermEvent_Sec_mean 2.84139
GrowthLifetimeBeforeTermEvent_Sec_SE 0.095539
GrowthLengthBeforeTermEvent_Mic_mean 2.69103
GrowthLengthBeforeTermEvent_Mic_SE 0.109961
term_freq_time_mean 0.465775
term_freq_time_SE 0.00857741
term_freq_length_mean 0.741346
term_freq_length_SE 0.0317449
ratio_preFgapVel2preTermVel 0.932114
ratio_preBgapVel2preTermVel NaN
ratio_preFgapVel2preBgapVel NaN
ratio_preFgapLife2preTermLife 1.03697
ratio_preBgapLife2preTermLife NaN
ratio_preFgapLife2preBgapLife NaN
ratio_preFgapDisp2preTermDisp NaN
ratio_preFgapDisp2preBgapDisp NaN
percentTimeGrowth 83.8308
percentTimeFgap 16.1692
percentTimeBgap 0
percentGapsForward 100
percentGapsBackward 0
percentGrowthLinkedBackward NaN
percentGrowthLinkedForward 14.359
percentGrowthTerminal 81.3675
```

```
percentGrowthLinkedUndefinedGap 4.2735
ratio_TotalTracks2NumGrowthSubtracks 0.817094
NumTracksWithfGap2TotalTracks_Per 13.8075
NumTracksWithbGap2TotalTracks_Per 0
NumTracksWithGap2TotalTracks_Per 13.8075
ratio_Compound2SingleTracks 0.177835
VelGrowthInCompTrack_mean_MicPerMin 60.7956
VelGrowthInSingleTrack_mean_MicPerMin 63.9995
LifeGrowthInCompTrack_mean_Sec 2.93182
LifeGrowthInSingleTrack_mean_Sec 2.83119
DispGrowthInCompTrack_mean_Mic 2.664
DispGrowthInSingleTrack_mean_Mic 2.71412
ratio_VelGrowthComp2Single 0.949939
ratio_LifeGrowthComp2Single 1.03554
ratio_DispGrowthComp2Single 0.981534
meanNumFgapsInMultTrackTraj 1.25758
avgIndivPercentTimeFgap 40.7505
avgIndivPercentTimeBgap NaN
meanNumBgapsInMultTrackTraj NaN
growth_speed_mean_IncludeAllPause 59.3802
growth_speed_median_IncludeAllPause 44.1005
growth_lifetime_mean_IncludeAllPause 4.036
growth_lifetime_median_IncludeAllPause 2.5
dynamicity 38.8764
avgComLatSec 0.0710312
numNucleationEvents 473
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN
```

Dati Anscombe-Wiener

```
medNNdistWithinFrameMic 1.37413
nGrowths 278
growth_speed_median 36.5992
growth_speed_mean 43.4349
growth_speed_std 26.8798
growth_lifetime_median 2
growth_lifetime_mean 2.57554
growth_lifetime_std 1.46919
growth_length_median 1.40305
growth_length_mean 1.74293
growth_length_std 1.16567
nFgaps 21
fgap_speed_median -5.38715
fgap_speed_mean -4.05805
fgap_speed_std 20.4186
fgap_lifetime_median 3.5
```

```
fgap_lifetime_mean 3.57143
fgap_lifetime_std 1.81167
fgap_length_median -0.346524
fgap_length_mean -0.132257
fgap_length_std 1.17365
GrowthSpeedBeforeFgap_MicPerMin_mean 49.68
GrowthSpeedBeforeFgap_MicPerMin_SE 6.98182
GrowthLifetimeBeforeFgap_Sec_mean 2.15909
GrowthLifetimeBeforeFgap_Sec_SE 0.184304
GrowthLengthBeforeFgap_Mic_mean 1.64682
GrowthLengthBeforeFgap_Mic_SE 0.209562
fgap_freq_time_mean 0.521429
fgap_freq_time_SE 0.0336229
fgap_freq_length_mean 0.884524
fgap_freq_length_SE 0.127338
nBgaps 1
bgap_speed_median -75.7633
bgap_speed_mean -75.7633
bgap_speed_std 0
bgap_lifetime_median 1
bgap_lifetime_mean 1
bgap_lifetime_std 0
bgap_length_median -1.26272
bgap_length_mean -1.26272
bgap_length_std 0
GrowthSpeedBeforeBgap_MicPerMin_mean 96.4472
GrowthSpeedBeforeBgap_MicPerMin_SE 0
GrowthLifetimeBeforeBgap_Sec_mean 2
GrowthLifetimeBeforeBgap_Sec_SE 0
GrowthLengthBeforeBgap_Mic_mean 3.21491
GrowthLengthBeforeBgap_Mic_SE 0
bgap_freq_time_mean 0.5
bgap_freq_time_SE 0
bgap_freq_length_mean 0.311051
bgap_freq_length_SE 0
nGrowthTermEvents 244
GrowthSpeedBeforeTermEvent_MicPerMin_mean 42.9913
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.7016
GrowthLifetimeBeforeTermEvent_Sec_mean 2.6168
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0980414
GrowthLengthBeforeTermEvent_Mic_mean 1.75806
GrowthLengthBeforeTermEvent_Mic_SE 0.0768178
term_freq_time_mean 0.475025
term_freq_time_SE 0.0115428
term_freq_length_mean 0.90238
term_freq_length_SE 0.0426428
ratio_preFgapVel2preTermVel 1.15558
ratio_preBgapVel2preTermVel 2.24341
ratio_preFgapVel2preBgapVel 0.515101
```

ratio_preFgapLife2preTermLife 0.825087
ratio_preBgapLife2preTermLife 0.764291
ratio_preFgapLife2preBgapLife 1.07955
ratio_preFgapDisp2preTermDisp 1.82867
ratio_preFgapDisp2preBgapDisp 0.512246
percentTimeGrowth 90.404
percentTimeFgap 9.4697
percentTimeBgap 0.126263
percentGapsForward 95.4545
percentGapsBackward 4.54545
percentGrowthLinkedBackward 0.359712
percentGrowthLinkedForward 7.91367
percentGrowthTerminal 87.7698
percentGrowthLinkedUndefinedGap 3.95683
ratio_TotalTracks2NumGrowthSubtracks 0.881295
NumTracksWithfGap2TotalTracks_Per 7.7551
NumTracksWithbGap2TotalTracks_Per 0.408163
NumTracksWithGap2TotalTracks_Per 7.7551
ratio_Compound2SingleTracks 0.0883721
VelGrowthInCompTrack_mean_MicPerMin 50.6014
VelGrowthInSingleTrack_mean_MicPerMin 42.6701
LifeGrowthInCompTrack_mean_Sec 2.63415
LifeGrowthInSingleTrack_mean_Sec 2.56279
DispGrowthInCompTrack_mean_Mic 2.03171
DispGrowthInSingleTrack_mean_Mic 1.71019
ratio_VelGrowthComp2Single 1.18588
ratio_LifeGrowthComp2Single 1.02784
ratio_DispGrowthComp2Single 1.188
meanNumFgapsInMultTrackTraj 1.10526
avgIndivPercentTimeFgap 38.4933
meanNumBgapsInMultTrackTraj 1
avgIndivPercentTimeBgap 14.2857
growth_speed_mean_IncludeAllPause 41.5633
growth_speed_median_IncludeAllPause 35.2641
growth_lifetime_mean_IncludeAllPause 3.07782
growth_lifetime_median_IncludeAllPause 2
dynamicity 32.8778
avgComLatSec -0.182696
numNucleationEvents 245
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN

Dati LOG-Wiener

medNNdistWithinFrameMic 1.44307
nGrowths 234
growth_speed_median 34.0037

```
growth_speed_mean 40.9974
growth_speed_std 26.624
growth_lifetime_median 2
growth_lifetime_mean 2.55983
growth_lifetime_std 1.38301
growth_length_median 1.34726
growth_length_mean 1.67239
growth_length_std 1.17077
nFgaps 21
fgap_speed_median 5.10443
fgap_speed_mean 2.43986
fgap_speed_std 11.191
fgap_lifetime_median 4
fgap_lifetime_mean 3.88095
fgap_lifetime_std 1.65759
fgap_length_median 0.155619
fgap_length_mean 0.0843379
fgap_length_std 0.704286
GrowthSpeedBeforeFgap_MicPerMin_mean 39.6621
GrowthSpeedBeforeFgap_MicPerMin_SE 5.35856
GrowthLifetimeBeforeFgap_Sec_mean 2.95238
GrowthLifetimeBeforeFgap_Sec_SE 0.412462
GrowthLengthBeforeFgap_Mic_mean 1.82927
GrowthLengthBeforeFgap_Mic_SE 0.246526
fgap_freq_time_mean 0.431581
fgap_freq_time_SE 0.0389608
fgap_freq_length_mean 0.854707
fgap_freq_length_SE 0.136363
nBgaps 5
bgap_speed_median -37.9471
bgap_speed_mean -52.0515
bgap_speed_std 34.0771
bgap_lifetime_median 2.5
bgap_lifetime_mean 2.2
bgap_lifetime_std 1.15109
bgap_length_median -1.37599
bgap_length_mean -1.40213
bgap_length_std 0.201145
GrowthSpeedBeforeBgap_MicPerMin_mean 83.8419
GrowthSpeedBeforeBgap_MicPerMin_SE 33.3786
GrowthLifetimeBeforeBgap_Sec_mean 2.1
GrowthLifetimeBeforeBgap_Sec_SE 0.244949
GrowthLengthBeforeBgap_Mic_mean 2.65488
GrowthLengthBeforeBgap_Mic_SE 0.756325
bgap_freq_time_mean 0.506667
bgap_freq_time_SE 0.0653197
bgap_freq_length_mean 0.524882
bgap_freq_length_SE 0.156701
nGrowthTermEvents 200
```

GrowthSpeedBeforeTermEvent_MicPerMin_mean 40.4032
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.72724
GrowthLifetimeBeforeTermEvent_Sec_mean 2.49
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0889611
GrowthLengthBeforeTermEvent_Mic_mean 1.61152
GrowthLengthBeforeTermEvent_Mic_SE 0.0740208
term_freq_time_mean 0.475801
term_freq_time_SE 0.011797
term_freq_length_mean 0.958785
term_freq_length_SE 0.0501418
ratio_preFgapVel2preTermVel 0.981658
ratio_preBgapVel2preTermVel 2.07513
ratio_preFgapVel2preBgapVel 0.473058
ratio_preFgapLife2preTermLife 1.1857
ratio_preBgapLife2preTermLife 0.843373
ratio_preFgapLife2preBgapLife 1.4059
ratio_preFgapDisp2preTermDisp 1.64744
ratio_preFgapDisp2preBgapDisp 0.689022
percentTimeGrowth 86.6233
percentTimeFgap 11.786
percentTimeBgap 1.59074
percentGapsForward 80.7692
percentGapsBackward 19.2308
percentGrowthLinkedBackward 2.13675
percentGrowthLinkedForward 8.97436
percentGrowthTerminal 85.4701
percentGrowthLinkedUndefinedGap 3.4188
ratio_TotalTracks2NumGrowthSubtracks 0.858974
NumTracksWithfGap2TotalTracks_Per 9.45274
NumTracksWithbGap2TotalTracks_Per 2.48756
NumTracksWithGap2TotalTracks_Per 10.4478
ratio_Compound2SingleTracks 0.128655
VelGrowthInCompTrack_mean_MicPerMin 45.7254
VelGrowthInSingleTrack_mean_MicPerMin 40.4368
LifeGrowthInCompTrack_mean_Sec 2.63542
LifeGrowthInSingleTrack_mean_Sec 2.4883
DispGrowthInCompTrack_mean_Mic 1.93486
DispGrowthInSingleTrack_mean_Mic 1.58415
ratio_VelGrowthComp2Single 1.13078
ratio_LifeGrowthComp2Single 1.05912
ratio_DispGrowthComp2Single 1.22139
meanNumFgapsInMultTrackTraj 1.10526
avgIndivPercentTimeFgap 39.4416
meanNumBgapsInMultTrackTraj 1
avgIndivPercentTimeBgap 22.4929
growth_speed_mean_IncludeAllPause 40.0546
growth_speed_median_IncludeAllPause 32.56
growth_lifetime_mean_IncludeAllPause 3.19484
growth_lifetime_median_IncludeAllPause 2

dynamicity 30.5222
avgComLatSec 0.123429
numNucleationEvents 200
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN

A.2 Statistiche di "Stack 6"

Dati originali

medNNdistWithinFrameMic 1.39506
nGrowths 538
growth_speed_median 26.3365
growth_speed_mean 33.1895
growth_speed_std 23.6023
growth_lifetime_median 2
growth_lifetime_mean 2.48885
growth_lifetime_std 1.78
growth_length_median 0.979431
growth_length_mean 1.3291
growth_length_std 1.14277
nFgaps 32
fgap_speed_median 7.16273
fgap_speed_mean 6.10055
fgap_speed_std 11.5449
fgap_lifetime_median 3.25
fgap_lifetime_mean 3.3125
fgap_lifetime_std 1.67885
fgap_length_median 0.310061
fgap_length_mean 0.382611
fgap_length_std 0.568079
GrowthSpeedBeforeFgap_MicPerMin_mean 34.1105
GrowthSpeedBeforeFgap_MicPerMin_SE 3.51933
GrowthLifetimeBeforeFgap_Sec_mean 2.51563
GrowthLifetimeBeforeFgap_Sec_SE 0.28174
GrowthLengthBeforeFgap_Mic_mean 1.37035
GrowthLengthBeforeFgap_Mic_SE 0.172351
fgap_freq_time_mean 0.481712
fgap_freq_time_SE 0.0292895
fgap_freq_length_mean 1.07126
fgap_freq_length_SE 0.108716
nBgaps 4
bgap_speed_median -40.3229
bgap_speed_mean -43.9873
bgap_speed_std 19.2594
bgap_lifetime_median 1.25

```
bgap_lifetime_mean 1.625
bgap_lifetime_std 0.946485
bgap_length_median -1.17943
bgap_length_mean -1.10415
bgap_length_std 0.49414
GrowthSpeedBeforeBgap_MicPerMin_mean 30.7317
GrowthSpeedBeforeBgap_MicPerMin_SE 5.90871
GrowthLifetimeBeforeBgap_Sec_mean 1.7
GrowthLifetimeBeforeBgap_Sec_SE 0.2
GrowthLengthBeforeBgap_Mic_mean 0.861048
GrowthLengthBeforeBgap_Mic_SE 0.164487
bgap_freq_time_mean 0.613333
bgap_freq_time_SE 0.0533333
bgap_freq_length_mean 1.37042
bgap_freq_length_SE 0.290831
nGrowthTermEvents 490
GrowthSpeedBeforeTermEvent_MicPerMin_mean 33.0523
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.08098
GrowthLifetimeBeforeTermEvent_Sec_mean 2.50612
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0820215
GrowthLengthBeforeTermEvent_Mic_mean 1.33348
GrowthLengthBeforeTermEvent_Mic_SE 0.0526222
term_freq_time_mean 0.491344
term_freq_time_SE 0.00766618
term_freq_length_mean 1.27877
term_freq_length_SE 0.0455371
ratio_preFgapVel2preTermVel 1.03202
ratio_preBgapVel2preTermVel 0.929791
ratio_preFgapVel2preBgapVel 1.10995
ratio_preFgapLife2preTermLife 1.00379
ratio_preBgapLife2preTermLife 0.678339
ratio_preFgapLife2preBgapLife 1.47978
ratio_preFgapDisp2preTermDisp 0.645713
ratio_preFgapDisp2preBgapDisp 1.5915
percentTimeGrowth 92.2494
percentTimeFgap 7.30279
percentTimeBgap 0.447813
percentGapsForward 88.8889
percentGapsBackward 11.1111
percentGrowthLinkedBackward 0.929368
percentGrowthLinkedForward 5.94796
percentGrowthTerminal 91.0781
percentGrowthLinkedUndefinedGap 2.04461
ratio_TotalTracks2NumGrowthSubtracks 0.912639
NumTracksWithfGap2TotalTracks_Per 6.10998
NumTracksWithbGap2TotalTracks_Per 0.814664
NumTracksWithGap2TotalTracks_Per 6.92464
ratio_Compound2SingleTracks 0.0835214
VelGrowthInCompTrack_mean_MicPerMin 31.8303
```

VelGrowthInSingleTrack_mean_MicPerMin 33.4069
LifeGrowthInCompTrack_mean_Sec 2.37162
LifeGrowthInSingleTrack_mean_Sec 2.52822
DispGrowthInCompTrack_mean_Mic 1.25324
DispGrowthInSingleTrack_mean_Mic 1.34933
ratio_VelGrowthComp2Single 0.952808
ratio_LifeGrowthComp2Single 0.938061
ratio_DispGrowthComp2Single 0.928788
meanNumFgapsInMultTrackTraj 1.06667
avgIndivPercentTimeFgap 38.9414
meanNumBgapsInMultTrackTraj 1
avgIndivPercentTimeBgap 30.6548
growth_speed_mean_IncludeAllPause 32.807
growth_speed_median_IncludeAllPause 26.2554
growth_lifetime_mean_IncludeAllPause 2.87599
growth_lifetime_median_IncludeAllPause 2
dynamicity 23.0493
avgComLatSec 0.691685
numNucleationEvents 488
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN

Dati Anscombe-Wiener

medNNdistWithinFrameMic 1.47888
nGrowths 512
growth_speed_median 26.9409
growth_speed_mean 32.4848
growth_speed_std 20.4155
growth_lifetime_median 2
growth_lifetime_mean 2.62891
growth_lifetime_std 1.75003
growth_length_median 1.04838
growth_length_mean 1.378
growth_length_std 1.07922
nFgaps 33
fgap_speed_median 3.24001
fgap_speed_mean -0.822902
fgap_speed_std 14.5591
fgap_lifetime_median 2.5
fgap_lifetime_mean 2.87879
fgap_lifetime_std 1.5614
fgap_length_median 0.133111
fgap_length_mean 0.0750003
fgap_length_std 0.668807
GrowthSpeedBeforeFgap_MicPerMin_mean 31.9839
GrowthSpeedBeforeFgap_MicPerMin_SE 2.92107

```
GrowthLifetimeBeforeFgap_Sec_mean 2.10606
GrowthLifetimeBeforeFgap_Sec_SE 0.138918
GrowthLengthBeforeFgap_Mic_mean 1.09237
GrowthLengthBeforeFgap_Mic_SE 0.1014
fgap_freq_time_mean 0.525565
fgap_freq_time_SE 0.0252708
fgap_freq_length_mean 1.27143
fgap_freq_length_SE 0.154652
nBgaps 1
bgap_speed_median -35.8601
bgap_speed_mean -35.8601
bgap_speed_std 0
bgap_lifetime_median 2
bgap_lifetime_mean 2
bgap_lifetime_std 0
bgap_length_median -1.19534
bgap_length_mean -1.19534
bgap_length_std 0
GrowthSpeedBeforeBgap_MicPerMin_mean 32.9638
GrowthSpeedBeforeBgap_MicPerMin_SE 0
GrowthLifetimeBeforeBgap_Sec_mean 1.5
GrowthLifetimeBeforeBgap_Sec_SE 0
GrowthLengthBeforeBgap_Mic_mean 0.824096
GrowthLengthBeforeBgap_Mic_SE 0
bgap_freq_time_mean 0.666667
bgap_freq_time_SE 0
bgap_freq_length_mean 1.21345
bgap_freq_length_SE 0
nGrowthTermEvents 465
GrowthSpeedBeforeTermEvent_MicPerMin_mean 32.4346
GrowthSpeedBeforeTermEvent_MicPerMin_SE 0.963783
GrowthLifetimeBeforeTermEvent_Sec_mean 2.68495
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0840329
GrowthLengthBeforeTermEvent_Mic_mean 1.4061
GrowthLengthBeforeTermEvent_Mic_SE 0.0517461
term_freq_time_mean 0.466461
term_freq_time_SE 0.00812452
term_freq_length_mean 1.20026
term_freq_length_SE 0.0427916
ratio_preFgapVel2preTermVel 0.986106
ratio_preBgapVel2preTermVel 1.01632
ratio_preFgapVel2preBgapVel 0.970272
ratio_preFgapLife2preTermLife 0.784396
ratio_preBgapLife2preTermLife 0.55867
ratio_preFgapLife2preBgapLife 1.40404
ratio_preFgapDisp2preTermDisp 0.586086
ratio_preFgapDisp2preBgapDisp 1.32553
percentTimeGrowth 93.2779
percentTimeFgap 6.58351
```

percentTimeBgap 0.1386
percentGapsForward 97.0588
percentGapsBackward 2.94118
percentGrowthLinkedBackward 0.195313
percentGrowthLinkedForward 6.44531
percentGrowthTerminal 90.8203
percentGrowthLinkedUndefinedGap 2.53906
ratio_TotalTracks2NumGrowthSubtracks 0.908203
NumTracksWithfGap2TotalTracks_Per 6.66667
NumTracksWithbGap2TotalTracks_Per 0.215054
NumTracksWithGap2TotalTracks_Per 6.66667
ratio_Compound2SingleTracks 0.0789474
VelGrowthInCompTrack_mean_MicPerMin 32.1572
VelGrowthInSingleTrack_mean_MicPerMin 32.0974
LifeGrowthInCompTrack_mean_Sec 2.28358
LifeGrowthInSingleTrack_mean_Sec 2.69737
DispGrowthInCompTrack_mean_Mic 1.19901
DispGrowthInSingleTrack_mean_Mic 1.39563
ratio_VelGrowthComp2Single 1.00186
ratio_LifeGrowthComp2Single 0.846596
ratio_DispGrowthComp2Single 0.859117
meanNumFgapsInMultTrackTraj 1.06452
avgIndivPercentTimeFgap 37.7671
meanNumBgapsInMultTrackTraj 1
avgIndivPercentTimeBgap 26.6667
growth_speed_mean_IncludeAllPause 31.8062
growth_speed_median_IncludeAllPause 26.2408
growth_lifetime_mean_IncludeAllPause 3.00835
growth_lifetime_median_IncludeAllPause 2
dynamicity 23.6356
avgComLatSec 0.138527
numNucleationEvents 461
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN

Dati LOG-Wiener

medNNdistWithinFrameMic 1.56254
nGrowths 443
growth_speed_median 27.1351
growth_speed_mean 32.4203
growth_speed_std 22.5935
growth_lifetime_median 2
growth_lifetime_mean 2.48758
growth_lifetime_std 1.56509
growth_length_median 0.970938
growth_length_mean 1.2953

```
growth_length_std 1.04892
nFgaps 20
fgap_speed_median 8.11156
fgap_speed_mean 4.27269
fgap_speed_std 13.3398
fgap_lifetime_median 2.25
fgap_lifetime_mean 2.775
fgap_lifetime_std 1.65016
fgap_length_median 0.289221
fgap_length_mean 0.148461
fgap_length_std 0.597125
GrowthSpeedBeforeFgap_MicPerMin_mean 32.7096
GrowthSpeedBeforeFgap_MicPerMin_SE 2.93986
GrowthLifetimeBeforeFgap_Sec_mean 2.72727
GrowthLifetimeBeforeFgap_Sec_SE 0.338265
GrowthLengthBeforeFgap_Mic_mean 1.34711
GrowthLengthBeforeFgap_Mic_SE 0.126104
fgap_freq_time_mean 0.449648
fgap_freq_time_SE 0.0376896
fgap_freq_length_mean 0.902886
fgap_freq_length_SE 0.0961133
nBgaps 3
bgap_speed_median -32.1009
bgap_speed_mean -31.0981
bgap_speed_std 2.35913
bgap_lifetime_median 1
bgap_lifetime_mean 1.5
bgap_lifetime_std 0.866025
bgap_length_median -0.535015
bgap_length_mean -0.791552
bgap_length_std 0.498661
GrowthSpeedBeforeBgap_MicPerMin_mean 35.5561
GrowthSpeedBeforeBgap_MicPerMin_SE 10.2877
GrowthLifetimeBeforeBgap_Sec_mean 1.625
GrowthLifetimeBeforeBgap_Sec_SE 0.125
GrowthLengthBeforeBgap_Mic_mean 0.938411
GrowthLengthBeforeBgap_Mic_SE 0.242629
bgap_freq_time_mean 0.625
bgap_freq_time_SE 0.0416667
bgap_freq_length_mean 1.24324
bgap_freq_length_SE 0.237313
nGrowthTermEvents 405
GrowthSpeedBeforeTermEvent_MicPerMin_mean 32.4559
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.1529
GrowthLifetimeBeforeTermEvent_Sec_mean 2.49506
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0783528
GrowthLengthBeforeTermEvent_Mic_mean 1.30191
GrowthLengthBeforeTermEvent_Mic_SE 0.0534367
term_freq_time_mean 0.489655
```

```
term_freq_time_SE 0.00857542
term_freq_length_mean 1.26573
term_freq_length_SE 0.0490645
ratio_preFgapVel2preTermVel 1.00782
ratio_preBgapVel2preTermVel 1.09552
ratio_preFgapVel2preBgapVel 0.919942
ratio_preFgapLife2preTermLife 1.09307
ratio_preBgapLife2preTermLife 0.651286
ratio_preFgapLife2preBgapLife 1.67832
ratio_preFgapDisp2preTermDisp 0.720797
ratio_preFgapDisp2preBgapDisp 1.43552
percentTimeGrowth 94.8365
percentTimeFgap 4.77625
percentTimeBgap 0.387263
percentGapsForward 86.9565
percentGapsBackward 13.0435
percentGrowthLinkedBackward 0.902935
percentGrowthLinkedForward 4.96614
percentGrowthTerminal 91.4221
percentGrowthLinkedUndefinedGap 2.7088
ratio_TotalTracks2NumGrowthSubtracks 0.920993
NumTracksWithfGap2TotalTracks_Per 4.65686
NumTracksWithbGap2TotalTracks_Per 0.735294
NumTracksWithGap2TotalTracks_Per 4.90196
ratio_Compound2SingleTracks 0.0672043
VelGrowthInCompTrack_mean_MicPerMin 32.2512
VelGrowthInSingleTrack_mean_MicPerMin 32.5632
LifeGrowthInCompTrack_mean_Sec 2.73958
LifeGrowthInSingleTrack_mean_Sec 2.47581
DispGrowthInCompTrack_mean_Mic 1.34709
DispGrowthInSingleTrack_mean_Mic 1.29667
ratio_VelGrowthComp2Single 0.990416
ratio_LifeGrowthComp2Single 1.10654
ratio_DispGrowthComp2Single 1.03889
meanNumFgapsInMultTrackTraj 1.05263
avgIndivPercentTimeFgap 31.6997
meanNumBgapsInMultTrackTraj 1
avgIndivPercentTimeBgap 14.7112
growth_speed_mean_IncludeAllPause 32.0282
growth_speed_median_IncludeAllPause 27.0114
growth_lifetime_mean_IncludeAllPause 2.73641
growth_lifetime_median_IncludeAllPause 2
dynamicity 23.314
avgComLatSec 0.274755
numNucleationEvents 405
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN
```

A.3 Statistiche di “Stack 11”

Dati originali

```
medNNdistWithinFrameMic 1.70038
nGrowths 401
growth_speed_median 38.0075
growth_speed_mean 55.5006
growth_speed_std 43.0401
growth_lifetime_median 2
growth_lifetime_mean 2.59726
growth_lifetime_std 2.12262
growth_length_median 1.54187
growth_length_mean 2.07829
growth_length_std 1.69646
nFgaps 33
fgap_speed_median 6.10128
fgap_speed_mean 3.78112
fgap_speed_std 27.0652
fgap_lifetime_median 3.5
fgap_lifetime_mean 3.62121
fgap_lifetime_std 1.72767
fgap_length_median 0.494007
fgap_length_mean 0.347784
fgap_length_std 1.25412
GrowthSpeedBeforeFgap_MicPerMin_mean 60.1213
GrowthSpeedBeforeFgap_MicPerMin_SE 6.87946
GrowthLifetimeBeforeFgap_Sec_mean 2.25758
GrowthLifetimeBeforeFgap_Sec_SE 0.276204
GrowthLengthBeforeFgap_Mic_mean 1.99184
GrowthLengthBeforeFgap_Mic_SE 0.217441
fgap_freq_time_mean 0.536018
fgap_freq_time_SE 0.028147
fgap_freq_length_mean 0.71976
fgap_freq_length_SE 0.075289
nBgaps 0
bgap_speed_median NaN
bgap_speed_mean NaN
bgap_speed_std NaN
bgap_lifetime_median NaN
bgap_lifetime_mean NaN
bgap_lifetime_std NaN
bgap_length_median NaN
bgap_length_mean NaN
bgap_length_std NaN
bgap_freq_time_mean NaN
bgap_freq_length_mean NaN
bgap_freq_time_SE NaN
bgap_freq_length_SE NaN
```

GrowthSpeedBeforeBgap_MicPerMin_mean NaN
GrowthSpeedBeforeBgap_MicPerMin_SE NaN
GrowthLifetimeBeforeBgap_Sec_mean NaN
GrowthLifetimeBeforeBgap_Sec_SE NaN
GrowthLengthBeforeBgap_Mic_mean NaN
GrowthLengthBeforeBgap_Mic_SE NaN
nGrowthTermEvents 361
GrowthSpeedBeforeTermEvent_MicPerMin_mean 54.8854
GrowthSpeedBeforeTermEvent_MicPerMin_SE 2.2961
GrowthLifetimeBeforeTermEvent_Sec_mean 2.63435
GrowthLifetimeBeforeTermEvent_Sec_SE 0.114581
GrowthLengthBeforeTermEvent_Mic_mean 2.07971
GrowthLengthBeforeTermEvent_Mic_SE 0.0915157
term_freq_time_mean 0.498731
term_freq_time_SE 0.00949083
term_freq_length_mean 0.844018
term_freq_length_SE 0.0337697
ratio_preFgapVel2preTermVel 1.0954
ratio_preBgapVel2preTermVel NaN
ratio_preFgapVel2preBgapVel NaN
ratio_preFgapLife2preTermLife 0.856977
ratio_preBgapLife2preTermLife NaN
ratio_preFgapLife2preBgapLife NaN
ratio_preFgapDisp2preTermDisp NaN
ratio_preFgapDisp2preBgapDisp NaN
percentTimeGrowth 89.7071
percentTimeFgap 10.2929
percentTimeBgap 0
percentGapsForward 100
percentGapsBackward 0
percentGrowthLinkedBackward NaN
percentGrowthLinkedForward 8.22943
percentGrowthTerminal 90.0249
percentGrowthLinkedUndefinedGap 1.74564
ratio_TotalTracks2NumGrowthSubtracks 0.900249
NumTracksWithfGap2TotalTracks_Per 8.31025
NumTracksWithbGap2TotalTracks_Per 0
NumTracksWithGap2TotalTracks_Per 8.31025
ratio_Compound2SingleTracks 0.0959752
VelGrowthInCompTrack_mean_MicPerMin 61.8396
VelGrowthInSingleTrack_mean_MicPerMin 54.3357
LifeGrowthInCompTrack_mean_Sec 2.65625
LifeGrowthInSingleTrack_mean_Sec 2.57895
DispGrowthInCompTrack_mean_Mic 2.34034
DispGrowthInSingleTrack_mean_Mic 2.02646
ratio_VelGrowthComp2Single 1.1381
ratio_LifeGrowthComp2Single 1.02997
ratio_DispGrowthComp2Single 1.15489
meanNumFgapsInMultTrackTraj 1.1

avgIndivPercentTimeFgap 39.9165
avgIndivPercentTimeBgap NaN
meanNumBgapsInMultTrackTraj NaN
growth_speed_mean_IncludeAllPause 53.3487
growth_speed_median_IncludeAllPause 36.2599
growth_lifetime_mean_IncludeAllPause 3.15489
growth_lifetime_median_IncludeAllPause 2
dynamicity 38.236
avgComLatSec 0.375978
numNucleationEvents 359
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN

Dati Anscombe-Wiener

medNNdistWithinFrameMic 1.80911
nGrowths 373
growth_speed_median 35.7049
growth_speed_mean 53.7422
growth_speed_std 43.9082
growth_lifetime_median 2
growth_lifetime_mean 2.58713
growth_lifetime_std 1.63909
growth_length_median 1.55163
growth_length_mean 2.07802
growth_length_std 1.68593
nFgaps 25
fgap_speed_median -5.57147
fgap_speed_mean -5.10276
fgap_speed_std 19.7343
fgap_lifetime_median 3
fgap_lifetime_mean 3.02
fgap_lifetime_std 1.81131
fgap_length_median -0.336691
fgap_length_mean -0.182254
fgap_length_std 0.781633
GrowthSpeedBeforeFgap_MicPerMin_mean 49.8958
GrowthSpeedBeforeFgap_MicPerMin_SE 6.62461
GrowthLifetimeBeforeFgap_Sec_mean 2.6
GrowthLifetimeBeforeFgap_Sec_SE 0.281366
GrowthLengthBeforeFgap_Mic_mean 1.95206
GrowthLengthBeforeFgap_Mic_SE 0.253258
fgap_freq_time_mean 0.471368
fgap_freq_time_SE 0.0366224
fgap_freq_length_mean 0.697835
fgap_freq_length_SE 0.0760803
nBgaps 3

```
bgap_speed_median -69.4715
bgap_speed_mean -85.1291
bgap_speed_std 28.7127
bgap_lifetime_median 1.5
bgap_lifetime_mean 1.5
bgap_lifetime_std 0.5
bgap_length_median -1.97112
bgap_length_mean -1.99268
bgap_length_std 0.312808
GrowthSpeedBeforeBgap_MicPerMin_mean 48.4046
GrowthSpeedBeforeBgap_MicPerMin_SE 15.4054
GrowthLifetimeBeforeBgap_Sec_mean 2.16667
GrowthLifetimeBeforeBgap_Sec_SE 0.666667
GrowthLengthBeforeBgap_Mic_mean 1.4913
GrowthLengthBeforeBgap_Mic_SE 0.255001
bgap_freq_time_mean 0.539683
bgap_freq_time_SE 0.126984
bgap_freq_length_mean 0.712846
bgap_freq_length_SE 0.125497
nGrowthTermEvents 332
GrowthSpeedBeforeTermEvent_MicPerMin_mean 53.6452
GrowthSpeedBeforeTermEvent_MicPerMin_SE 2.45416
GrowthLifetimeBeforeTermEvent_Sec_mean 2.5738
GrowthLifetimeBeforeTermEvent_Sec_SE 0.089076
GrowthLengthBeforeTermEvent_Mic_mean 2.07804
GrowthLengthBeforeTermEvent_Mic_SE 0.0952697
term_freq_time_mean 0.478614
term_freq_time_SE 0.00941049
term_freq_length_mean 0.858471
term_freq_length_SE 0.0380524
ratio_preFgapVel2preTermVel 0.930109
ratio_preBgapVel2preTermVel 0.90231
ratio_preFgapVel2preBgapVel 1.03081
ratio_preFgapLife2preTermLife 1.01018
ratio_preBgapLife2preTermLife 0.841818
ratio_preFgapLife2preBgapLife 1.2
ratio_preFgapDisp2preTermDisp 0.717649
ratio_preFgapDisp2preBgapDisp 1.30897
percentTimeGrowth 92.3445
percentTimeFgap 7.22488
percentTimeBgap 0.430622
percentGapsForward 89.2857
percentGapsBackward 10.7143
percentGrowthLinkedBackward 0.80429
percentGrowthLinkedForward 6.70241
percentGrowthTerminal 89.008
percentGrowthLinkedUndefinedGap 3.48525
ratio_TotalTracks2NumGrowthSubtracks 0.89008
NumTracksWithfGap2TotalTracks_Per 6.92771
```

NumTracksWithbGap2TotalTracks_Per 0.903614
NumTracksWithGap2TotalTracks_Per 7.83133
ratio_Compound2SingleTracks 0.0887372
VelGrowthInCompTrack_mean_MicPerMin 45.9402
VelGrowthInSingleTrack_mean_MicPerMin 54.7933
LifeGrowthInCompTrack_mean_Sec 2.57407
LifeGrowthInSingleTrack_mean_Sec 2.58874
DispGrowthInCompTrack_mean_Mic 1.79036
DispGrowthInSingleTrack_mean_Mic 2.11668
ratio_VelGrowthComp2Single 0.838428
ratio_LifeGrowthComp2Single 0.994336
ratio_DispGrowthComp2Single 0.845835
meanNumFgapsInMultTrackTraj 1.08696
avgIndivPercentTimeFgap 36.7567
meanNumBgapsInMultTrackTraj 1
avgIndivPercentTimeBgap 25.1282
growth_speed_mean_IncludeAllPause 53.0784
growth_speed_median_IncludeAllPause 33.8341
growth_lifetime_mean_IncludeAllPause 2.98994
growth_lifetime_median_IncludeAllPause 2
dynamicity 32.5541
avgComLatSec -0.203476
numNucleationEvents 331
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN

Dati LOG-Wiener

medNNdistWithinFrameMic 1.92954
nGrowths 328
growth_speed_median 35.4738
growth_speed_mean 52.509
growth_speed_std 43.1417
growth_lifetime_median 2
growth_lifetime_mean 2.6753
growth_lifetime_std 1.66422
growth_length_median 1.52885
growth_length_mean 2.05027
growth_length_std 1.55032
nFgaps 21
fgap_speed_median 4.4439
fgap_speed_mean 7.8589
fgap_speed_std 26.4451
fgap_lifetime_median 3.5
fgap_lifetime_mean 3.45238
fgap_lifetime_std 1.42219
fgap_length_median 0.291159

```
fgap_length_mean 0.617323
fgap_length_std 1.56622
GrowthSpeedBeforeFgap_MicPerMin_mean 69.5926
GrowthSpeedBeforeFgap_MicPerMin_SE 11.7028
GrowthLifetimeBeforeFgap_Sec_mean 2.30435
GrowthLifetimeBeforeFgap_Sec_SE 0.246996
GrowthLengthBeforeFgap_Mic_mean 2.3464
GrowthLengthBeforeFgap_Mic_SE 0.353599
fgap_freq_time_mean 0.511149
fgap_freq_time_SE 0.0349827
fgap_freq_length_mean 0.649883
fgap_freq_length_SE 0.0771353
nBgaps 1
bgap_speed_median -90.4734
bgap_speed_mean -90.4734
bgap_speed_std 0
bgap_lifetime_median 1
bgap_lifetime_mean 1
bgap_lifetime_std 0
bgap_length_median -1.50789
bgap_length_mean -1.50789
bgap_length_std 0
GrowthSpeedBeforeBgap_MicPerMin_mean 27.096
GrowthSpeedBeforeBgap_MicPerMin_SE 0
GrowthLifetimeBeforeBgap_Sec_mean 5.5
GrowthLifetimeBeforeBgap_Sec_SE 0
GrowthLengthBeforeBgap_Mic_mean 2.4838
GrowthLengthBeforeBgap_Mic_SE 0
bgap_freq_time_mean 0.181818
bgap_freq_time_SE 0
bgap_freq_length_mean 0.402609
bgap_freq_length_SE 0
nGrowthTermEvents 294
GrowthSpeedBeforeTermEvent_MicPerMin_mean 51.6195
GrowthSpeedBeforeTermEvent_MicPerMin_SE 2.47021
GrowthLifetimeBeforeTermEvent_Sec_mean 2.65986
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0957337
GrowthLengthBeforeTermEvent_Mic_mean 2.00943
GrowthLengthBeforeTermEvent_Mic_SE 0.0887498
term_freq_time_mean 0.467389
term_freq_time_SE 0.0101274
term_freq_length_mean 0.81051
term_freq_length_SE 0.0363561
ratio_preFgapVel2preTermVel 1.34818
ratio_preBgapVel2preTermVel 0.524917
ratio_preFgapVel2preBgapVel 2.56837
ratio_preFgapLife2preTermLife 0.86634
ratio_preBgapLife2preTermLife 2.06777
ratio_preFgapLife2preBgapLife 0.418972
```

```
ratio_preFgapDisp2preTermDisp 1.23607
ratio_preFgapDisp2preBgapDisp 0.944681
percentTimeGrowth 92.2713
percentTimeFgap 7.62355
percentTimeBgap 0.105152
percentGapsForward 95.4545
percentGapsBackward 4.54545
percentGrowthLinkedBackward 0.304878
percentGrowthLinkedForward 7.0122
percentGrowthTerminal 89.6341
percentGrowthLinkedUndefinedGap 3.04878
ratio_TotalTracks2NumGrowthSubtracks 0.905488
NumTracksWithfGap2TotalTracks_Per 7.07071
NumTracksWithbGap2TotalTracks_Per 0.3367
NumTracksWithGap2TotalTracks_Per 7.40741
ratio_Compound2SingleTracks 0.0961538
VelGrowthInCompTrack_mean_MicPerMin 58.1833
VelGrowthInSingleTrack_mean_MicPerMin 52.4009
LifeGrowthInCompTrack_mean_Sec 2.38298
LifeGrowthInSingleTrack_mean_Sec 2.67115
DispGrowthInCompTrack_mean_Mic 2.0467
DispGrowthInSingleTrack_mean_Mic 2.03896
ratio_VelGrowthComp2Single 1.11035
ratio_LifeGrowthComp2Single 0.892116
ratio_DispGrowthComp2Single 1.00379
meanNumFgapsInMultTrackTraj 1
avgIndivPercentTimeFgap 41.9185
meanNumBgapsInMultTrackTraj 1
avgIndivPercentTimeBgap 6.89655
growth_speed_mean_IncludeAllPause 50.7024
growth_speed_median_IncludeAllPause 34.4028
growth_lifetime_mean_IncludeAllPause 3.09446
growth_lifetime_median_IncludeAllPause 2
dynamicity 40.0063
avgComLatSec 0.705391
numNucleationEvents 294
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN
```

A.4 Statistiche di “Stack 14”

Dati originali

```
medNNdistWithinFrameMic 1.44221
nGrowths 412
growth_speed_median 33.1149
```

```
growth_speed_mean 41.3639
growth_speed_std 27.846
growth_lifetime_median 2
growth_lifetime_mean 2.62379
growth_lifetime_std 1.54745
growth_length_median 1.29134
growth_length_mean 1.71704
growth_length_std 1.264
nFgaps 51
fgap_speed_median -1.73944
fgap_speed_mean -4.54443
fgap_speed_std 22.6086
fgap_lifetime_median 3
fgap_lifetime_mean 3.08824
fgap_lifetime_std 1.8404
fgap_length_median -0.115963
fgap_length_mean -0.078572
fgap_length_std 0.994801
GrowthSpeedBeforeFgap_MicPerMin_mean 50.8882
GrowthSpeedBeforeFgap_MicPerMin_SE 3.77893
GrowthLifetimeBeforeFgap_Sec_mean 2.75926
GrowthLifetimeBeforeFgap_Sec_SE 0.277171
GrowthLengthBeforeFgap_Mic_mean 2.14875
GrowthLengthBeforeFgap_Mic_SE 0.186515
fgap_freq_time_mean 0.458956
fgap_freq_time_SE 0.0233504
fgap_freq_length_mean 0.721632
fgap_freq_length_SE 0.0776123
nBgaps 0
bgap_speed_median NaN
bgap_speed_mean NaN
bgap_speed_std NaN
bgap_lifetime_median NaN
bgap_lifetime_mean NaN
bgap_lifetime_std NaN
bgap_length_median NaN
bgap_length_mean NaN
bgap_length_std NaN
bgap_freq_time_mean NaN
bgap_freq_length_mean NaN
bgap_freq_time_SE NaN
bgap_freq_length_SE NaN
GrowthSpeedBeforeBgap_MicPerMin_mean NaN
GrowthSpeedBeforeBgap_MicPerMin_SE NaN
GrowthLifetimeBeforeBgap_Sec_mean NaN
GrowthLifetimeBeforeBgap_Sec_SE NaN
GrowthLengthBeforeBgap_Mic_mean NaN
GrowthLengthBeforeBgap_Mic_SE NaN
nGrowthTermEvents 344
```

```
GrowthSpeedBeforeTermEvent_MicPerMin_mean 39.9176
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.50474
GrowthLifetimeBeforeTermEvent_Sec_mean 2.62209
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0796136
GrowthLengthBeforeTermEvent_Mic_mean 1.66512
GrowthLengthBeforeTermEvent_Mic_SE 0.0676503
term_freq_time_mean 0.466332
term_freq_time_SE 0.00935367
term_freq_length_mean 0.994887
term_freq_length_SE 0.0412563
ratio_preFgapVel2preTermVel 1.27483
ratio_preBgapVel2preTermVel NaN
ratio_preFgapVel2preBgapVel NaN
ratio_preFgapLife2preTermLife 1.05231
ratio_preBgapLife2preTermLife NaN
ratio_preFgapLife2preBgapLife NaN
ratio_preFgapDisp2preTermDisp NaN
ratio_preFgapDisp2preBgapDisp NaN
percentTimeGrowth 87.283
percentTimeFgap 12.717
percentTimeBgap 0
percentGapsForward 100
percentGapsBackward 0
percentGrowthLinkedBackward NaN
percentGrowthLinkedForward 13.1068
percentGrowthTerminal 83.4951
percentGrowthLinkedUndefinedGap 3.39806
ratio_TotalTracks2NumGrowthSubtracks 0.842233
NumTracksWithfGap2TotalTracks_Per 11.5274
NumTracksWithbGap2TotalTracks_Per 0
NumTracksWithGap2TotalTracks_Per 11.5274
ratio_Compound2SingleTracks 0.14433
VelGrowthInCompTrack_mean_MicPerMin 44.9165
VelGrowthInSingleTrack_mean_MicPerMin 40.3316
LifeGrowthInCompTrack_mean_Sec 2.82979
LifeGrowthInSingleTrack_mean_Sec 2.60137
DispGrowthInCompTrack_mean_Mic 2.00964
DispGrowthInSingleTrack_mean_Mic 1.65511
ratio_VelGrowthComp2Single 1.11368
ratio_LifeGrowthComp2Single 1.0878
ratio_DispGrowthComp2Single 1.2142
meanNumFgapsInMultTrackTraj 1.275
avgIndivPercentTimeFgap 35.2839
avgIndivPercentTimeBgap NaN
meanNumBgapsInMultTrackTraj NaN
growth_speed_mean_IncludeAllPause 39.0313
growth_speed_median_IncludeAllPause 31.1306
growth_lifetime_mean_IncludeAllPause 3.45278
growth_lifetime_median_IncludeAllPause 2.5
```

dynamicity 31.4589
avgComLatSec -0.113972
numNucleationEvents 347
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN

Dati Anscombe-Wiener

medNNdistWithinFrameMic 1.5356
nGrowths 378
growth_speed_median 30.799
growth_speed_mean 39.4454
growth_speed_std 28.2964
growth_lifetime_median 2
growth_lifetime_mean 2.66402
growth_lifetime_std 1.81968
growth_length_median 1.23385
growth_length_mean 1.64191
growth_length_std 1.29076
nFgaps 41
fgap_speed_median -6.11372
fgap_speed_mean -3.31208
fgap_speed_std 12.6437
fgap_lifetime_median 4
fgap_lifetime_mean 3.95122
fgap_lifetime_std 1.87018
fgap_length_median -0.342802
fgap_length_mean -0.292881
fgap_length_std 0.690533
GrowthSpeedBeforeFgap_MicPerMin_mean 40.9108
GrowthSpeedBeforeFgap_MicPerMin_SE 3.95245
GrowthLifetimeBeforeFgap_Sec_mean 3.22619
GrowthLifetimeBeforeFgap_Sec_SE 0.580661
GrowthLengthBeforeFgap_Mic_mean 1.85433
GrowthLengthBeforeFgap_Mic_SE 0.243535
fgap_freq_time_mean 0.468791
fgap_freq_time_SE 0.029642
fgap_freq_length_mean 0.94684
fgap_freq_length_SE 0.126172
nBgaps 0
bgap_speed_median NaN
bgap_speed_mean NaN
bgap_speed_std NaN
bgap_lifetime_median NaN
bgap_lifetime_mean NaN
bgap_lifetime_std NaN
bgap_length_median NaN

bgap_length_mean NaN
bgap_length_std NaN
bgap_freq_time_mean NaN
bgap_freq_length_mean NaN
bgap_freq_time_SE NaN
bgap_freq_length_SE NaN
GrowthSpeedBeforeBgap_MicPerMin_mean NaN
GrowthSpeedBeforeBgap_MicPerMin_SE NaN
GrowthLifetimeBeforeBgap_Sec_mean NaN
GrowthLifetimeBeforeBgap_Sec_SE NaN
GrowthLengthBeforeBgap_Mic_mean NaN
GrowthLengthBeforeBgap_Mic_SE NaN
nGrowthTermEvents 324
GrowthSpeedBeforeTermEvent_MicPerMin_mean 39.434
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.59959
GrowthLifetimeBeforeTermEvent_Sec_mean 2.58796
GrowthLifetimeBeforeTermEvent_Sec_SE 0.077233
GrowthLengthBeforeTermEvent_Mic_mean 1.62428
GrowthLengthBeforeTermEvent_Mic_SE 0.0701163
term_freq_time_mean 0.469985
term_freq_time_SE 0.00970992
term_freq_length_mean 1.03102
term_freq_length_SE 0.0446357
ratio_preFgapVel2preTermVel 1.03745
ratio_preBgapVel2preTermVel NaN
ratio_preFgapVel2preBgapVel NaN
ratio_preFgapLife2preTermLife 1.24661
ratio_preBgapLife2preTermLife NaN
ratio_preFgapLife2preBgapLife NaN
ratio_preFgapDisp2preTermDisp NaN
ratio_preFgapDisp2preBgapDisp NaN
percentTimeGrowth 86.142
percentTimeFgap 13.858
percentTimeBgap 0
percentGapsForward 100
percentGapsBackward 0
percentGrowthLinkedBackward NaN
percentGrowthLinkedForward 11.1111
percentGrowthTerminal 85.7143
percentGrowthLinkedUndefinedGap 3.1746
ratio_TotalTracks2NumGrowthSubtracks 0.859788
NumTracksWithfGap2TotalTracks_Per 12.3077
NumTracksWithbGap2TotalTracks_Per 0
NumTracksWithGap2TotalTracks_Per 12.3077
ratio_Compound2SingleTracks 0.149635
VelGrowthInCompTrack_mean_MicPerMin 38.4263
VelGrowthInSingleTrack_mean_MicPerMin 40.1034
LifeGrowthInCompTrack_mean_Sec 2.98171
LifeGrowthInSingleTrack_mean_Sec 2.54562

DispGrowthInCompTrack_mean_Mic 1.70697
DispGrowthInSingleTrack_mean_Mic 1.62087
ratio_VelGrowthComp2Single 0.958182
ratio_LifeGrowthComp2Single 1.17131
ratio_DispGrowthComp2Single 1.05311
meanNumFgapsInMultTrackTraj 1.025
avgIndivPercentTimeFgap 39.6218
avgIndivPercentTimeBgap NaN
meanNumBgapsInMultTrackTraj NaN
growth_speed_mean_IncludeAllPause 37.9277
growth_speed_median_IncludeAllPause 30.2307
growth_lifetime_mean_IncludeAllPause 3.46884
growth_lifetime_median_IncludeAllPause 2.5
dynamicity 24.4832
avgComLatSec -0.445498
numNucleationEvents 324
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN

Dati LOG-Wiener

medNNdistWithinFrameMic 1.62236
nGrowths 349
growth_speed_median 30.7705
growth_speed_mean 37.5244
growth_speed_std 22.6634
growth_lifetime_median 2
growth_lifetime_mean 2.54441
growth_lifetime_std 1.39484
growth_length_median 1.25913
growth_length_mean 1.50591
growth_length_std 1.01059
nFgaps 33
fgap_speed_median -6.15656
fgap_speed_mean -2.44486
fgap_speed_std 16.6524
fgap_lifetime_median 3
fgap_lifetime_mean 3.43939
fgap_lifetime_std 1.89472
fgap_length_median -0.308944
fgap_length_mean -0.187895
fgap_length_std 0.742987
GrowthSpeedBeforeFgap_MicPerMin_mean 41.0247
GrowthSpeedBeforeFgap_MicPerMin_SE 3.49515
GrowthLifetimeBeforeFgap_Sec_mean 3.06061
GrowthLifetimeBeforeFgap_Sec_SE 0.326945
GrowthLengthBeforeFgap_Mic_mean 1.88045
GrowthLengthBeforeFgap_Mic_SE 0.192747

```
fgap_freq_time_mean 0.433202
fgap_freq_time_SE 0.0340092
fgap_freq_length_mean 0.798561
fgap_freq_length_SE 0.126195
nBgaps 0
bgap_speed_median NaN
bgap_speed_mean NaN
bgap_speed_std NaN
bgap_lifetime_median NaN
bgap_lifetime_mean NaN
bgap_lifetime_std NaN
bgap_length_median NaN
bgap_length_mean NaN
bgap_length_std NaN
bgap_freq_time_mean NaN
bgap_freq_length_mean NaN
bgap_freq_time_SE NaN
bgap_freq_length_SE NaN
GrowthSpeedBeforeBgap_MicPerMin_mean NaN
GrowthSpeedBeforeBgap_MicPerMin_SE NaN
GrowthLifetimeBeforeBgap_Sec_mean NaN
GrowthLifetimeBeforeBgap_Sec_SE NaN
GrowthLengthBeforeBgap_Mic_mean NaN
GrowthLengthBeforeBgap_Mic_SE NaN
nGrowthTermEvents 306
GrowthSpeedBeforeTermEvent_MicPerMin_mean 37.3102
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.31799
GrowthLifetimeBeforeTermEvent_Sec_mean 2.5
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0768012
GrowthLengthBeforeTermEvent_Mic_mean 1.47607
GrowthLengthBeforeTermEvent_Mic_SE 0.0570634
term_freq_time_mean 0.479657
term_freq_time_SE 0.00958361
term_freq_length_mean 1.01064
term_freq_length_SE 0.0399966
ratio_preFgapVel2preTermVel 1.09956
ratio_preBgapVel2preTermVel NaN
ratio_preFgapVel2preBgapVel NaN
ratio_preFgapLife2preTermLife 1.22424
ratio_preBgapLife2preTermLife NaN
ratio_preFgapLife2preBgapLife NaN
ratio_preFgapDisp2preTermDisp NaN
ratio_preFgapDisp2preBgapDisp NaN
percentTimeGrowth 88.667
percentTimeFgap 11.333
percentTimeBgap 0
percentGapsForward 100
percentGapsBackward 0
percentGrowthLinkedBackward NaN
```

```
percentGrowthLinkedForward 9.45559
percentGrowthTerminal 87.6791
percentGrowthLinkedUndefinedGap 2.86533
ratio_TotalTracks2NumGrowthSubtracks 0.876791
NumTracksWithfGap2TotalTracks_Per 9.80392
NumTracksWithbGap2TotalTracks_Per 0
NumTracksWithGap2TotalTracks_Per 9.80392
ratio_Compound2SingleTracks 0.116541
VelGrowthInCompTrack_mean_MicPerMin 37.7183
VelGrowthInSingleTrack_mean_MicPerMin 37.7056
LifeGrowthInCompTrack_mean_Sec 2.8125
LifeGrowthInSingleTrack_mean_Sec 2.50188
DispGrowthInCompTrack_mean_Mic 1.68619
DispGrowthInSingleTrack_mean_Mic 1.4771
ratio_VelGrowthComp2Single 1.00033
ratio_LifeGrowthComp2Single 1.12415
ratio_DispGrowthComp2Single 1.14155
meanNumFgapsInMultTrackTraj 1.1
avgIndivPercentTimeFgap 34.7701
avgIndivPercentTimeBgap NaN
meanNumBgapsInMultTrackTraj NaN
growth_speed_mean_IncludeAllPause 36.2681
growth_speed_median_IncludeAllPause 29.5888
growth_lifetime_mean_IncludeAllPause 3.1693
growth_lifetime_median_IncludeAllPause 2
dynamicity 25.633
avgComLatSec -0.300437
numNucleationEvents 305
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN
```

A.5 Statistiche di “Stack 20”

Dati originali

```
medNNdistWithinFrameMic 1.30107
nGrowths 643
growth_speed_median 32.1669
growth_speed_mean 41.3924
growth_speed_std 28.4898
growth_lifetime_median 2
growth_lifetime_mean 2.49533
growth_lifetime_std 1.38555
growth_length_median 1.28807
growth_length_mean 1.62761
growth_length_std 1.2339
```

```
nFgaps 66
fgap_speed_median -3.1308
fgap_speed_mean -2.92095
fgap_speed_std 17.4362
fgap_lifetime_median 3
fgap_lifetime_mean 3.12121
fgap_lifetime_std 1.70533
fgap_length_median -0.13735
fgap_length_mean -0.0363935
fgap_length_std 0.807655
GrowthSpeedBeforeFgap_MicPerMin_mean 39.2559
GrowthSpeedBeforeFgap_MicPerMin_SE 2.85036
GrowthLifetimeBeforeFgap_Sec_mean 2.28788
GrowthLifetimeBeforeFgap_Sec_SE 0.110656
GrowthLengthBeforeFgap_Mic_mean 1.46079
GrowthLengthBeforeFgap_Mic_SE 0.128187
fgap_freq_time_mean 0.49614
fgap_freq_time_SE 0.0196237
fgap_freq_length_mean 1.04345
fgap_freq_length_SE 0.0951379
nBgaps 2
bgap_speed_median -62.78
bgap_speed_mean -62.78
bgap_speed_std 10.2397
bgap_lifetime_median 1.75
bgap_lifetime_mean 1.75
bgap_lifetime_std 0.353553
bgap_length_median -1.80092
bgap_length_mean -1.80092
bgap_length_std 0.0712759
GrowthSpeedBeforeBgap_MicPerMin_mean 50.0849
GrowthSpeedBeforeBgap_MicPerMin_SE 14.9811
GrowthLifetimeBeforeBgap_Sec_mean 5.33333
GrowthLifetimeBeforeBgap_Sec_SE 2.83333
GrowthLengthBeforeBgap_Mic_mean 3.19866
GrowthLengthBeforeBgap_Mic_SE 0.629443
bgap_freq_time_mean 0.29697
bgap_freq_time_SE 0.10303
bgap_freq_length_mean 0.339232
bgap_freq_length_SE 0.0689309
nGrowthTermEvents 561
GrowthSpeedBeforeTermEvent_MicPerMin_mean 41.7485
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.2341
GrowthLifetimeBeforeTermEvent_Sec_mean 2.50802
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0589628
GrowthLengthBeforeTermEvent_Mic_mean 1.64685
GrowthLengthBeforeTermEvent_Mic_SE 0.0531539
term_freq_time_mean 0.482088
term_freq_time_SE 0.00727238
```

term_freq_length_mean 1.01464
term_freq_length_SE 0.0344259
ratio_preFgapVel2preTermVel 0.940295
ratio_preBgapVel2preTermVel 1.19968
ratio_preFgapVel2preBgapVel 0.783787
ratio_preFgapLife2preTermLife 0.912225
ratio_preBgapLife2preTermLife 2.12651
ratio_preFgapLife2preBgapLife 0.428977
ratio_preFgapDisp2preTermDisp 1.94228
ratio_preFgapDisp2preBgapDisp 0.456689
percentTimeGrowth 88.4509
percentTimeFgap 11.3561
percentTimeBgap 0.192944
percentGapsForward 97.0588
percentGapsBackward 2.94118
percentGrowthLinkedBackward 0.466563
percentGrowthLinkedForward 10.2644
percentGrowthTerminal 87.2473
percentGrowthLinkedUndefinedGap 2.02177
ratio_TotalTracks2NumGrowthSubtracks 0.874028
NumTracksWithfGap2TotalTracks_Per 10.4982
NumTracksWithbGap2TotalTracks_Per 0.355872
NumTracksWithGap2TotalTracks_Per 10.6762
ratio_Compound2SingleTracks 0.127572
VelGrowthInCompTrack_mean_MicPerMin 38.9875
VelGrowthInSingleTrack_mean_MicPerMin 42.2423
LifeGrowthInCompTrack_mean_Sec 2.5229
LifeGrowthInSingleTrack_mean_Sec 2.49486
DispGrowthInCompTrack_mean_Mic 1.57021
DispGrowthInSingleTrack_mean_Mic 1.6564
ratio_VelGrowthComp2Single 0.922951
ratio_LifeGrowthComp2Single 1.01124
ratio_DispGrowthComp2Single 0.947964
meanNumFgapsInMultTrackTraj 1.11864
avgIndivPercentTimeFgap 38.2576
meanNumBgapsInMultTrackTraj 1
avgIndivPercentTimeBgap 10.088
growth_speed_mean_IncludeAllPause 40.299
growth_speed_median_IncludeAllPause 31.4243
growth_lifetime_mean_IncludeAllPause 3.13778
growth_lifetime_median_IncludeAllPause 2
dynamicity 27.2345
avgComLatSec -0.052754
numNucleationEvents 558
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN

Dati Anscombe-Wiener

```
medNNdistWithinFrameMic 1.41024
nGrowths 581
growth_speed_median 31.1726
growth_speed_mean 40.6586
growth_speed_std 29.9742
growth_lifetime_median 2
growth_lifetime_mean 2.5938
growth_lifetime_std 1.65292
growth_length_median 1.26375
growth_length_mean 1.63572
growth_length_std 1.26084
nFgaps 42
fgap_speed_median -6.81255
fgap_speed_mean -4.81466
fgap_speed_std 18.4971
fgap_lifetime_median 3.75
fgap_lifetime_mean 3.66667
fgap_lifetime_std 1.85007
fgap_length_median -0.326493
fgap_length_mean -0.144094
fgap_length_std 0.886129
GrowthSpeedBeforeFgap_MicPerMin_mean 39.4963
GrowthSpeedBeforeFgap_MicPerMin_SE 3.35651
GrowthLifetimeBeforeFgap_Sec_mean 2.68605
GrowthLifetimeBeforeFgap_Sec_SE 0.244555
GrowthLengthBeforeFgap_Mic_mean 1.72787
GrowthLengthBeforeFgap_Mic_SE 0.204667
fgap_freq_time_mean 0.459237
fgap_freq_time_SE 0.0267115
fgap_freq_length_mean 0.862142
fgap_freq_length_SE 0.0769152
nBgaps 0
bgap_speed_median NaN
bgap_speed_mean NaN
bgap_speed_std NaN
bgap_lifetime_median NaN
bgap_lifetime_mean NaN
bgap_lifetime_std NaN
bgap_length_median NaN
bgap_length_mean NaN
bgap_length_std NaN
bgap_freq_time_mean NaN
bgap_freq_length_mean NaN
bgap_freq_time_SE NaN
bgap_freq_length_SE NaN
GrowthSpeedBeforeBgap_MicPerMin_mean NaN
GrowthSpeedBeforeBgap_MicPerMin_SE NaN
```

```
GrowthLifetimeBeforeBgap_Sec_mean NaN
GrowthLifetimeBeforeBgap_Sec_SE NaN
GrowthLengthBeforeBgap_Mic_mean NaN
GrowthLengthBeforeBgap_Mic_SE NaN
nGrowthTermEvents 516
GrowthSpeedBeforeTermEvent_MicPerMin_mean 40.3739
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.34705
GrowthLifetimeBeforeTermEvent_Sec_mean 2.59012
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0733069
GrowthLengthBeforeTermEvent_Mic_mean 1.62007
GrowthLengthBeforeTermEvent_Mic_SE 0.0556666
term_freq_time_mean 0.481139
term_freq_time_SE 0.00767715
term_freq_length_mean 1.03842
term_freq_length_SE 0.0342457
ratio_preFgapVel2preTermVel 0.978263
ratio_preBgapVel2preTermVel NaN
ratio_preFgapVel2preBgapVel NaN
ratio_preFgapLife2preTermLife 1.03704
ratio_preBgapLife2preTermLife NaN
ratio_preFgapLife2preBgapLife NaN
ratio_preFgapDisp2preTermDisp NaN
ratio_preFgapDisp2preBgapDisp NaN
percentTimeGrowth 90.7285
percentTimeFgap 9.27152
percentTimeBgap 0
percentGapsForward 100
percentGapsBackward 0
percentGrowthLinkedBackward NaN
percentGrowthLinkedForward 7.40103
percentGrowthTerminal 88.8124
percentGrowthLinkedUndefinedGap 3.78657
ratio_TotalTracks2NumGrowthSubtracks 0.893287
NumTracksWithfGap2TotalTracks_Per 7.32177
NumTracksWithbGap2TotalTracks_Per 0
NumTracksWithGap2TotalTracks_Per 7.32177
ratio_Compound2SingleTracks 0.0851528
VelGrowthInCompTrack_mean_MicPerMin 41.5128
VelGrowthInSingleTrack_mean_MicPerMin 40.0045
LifeGrowthInCompTrack_mean_Sec 2.62963
LifeGrowthInSingleTrack_mean_Sec 2.57205
DispGrowthInCompTrack_mean_Mic 1.71081
DispGrowthInSingleTrack_mean_Mic 1.59901
ratio_VelGrowthComp2Single 1.0377
ratio_LifeGrowthComp2Single 1.02239
ratio_DispGrowthComp2Single 1.06992
meanNumFgapsInMultTrackTraj 1.10526
avgIndivPercentTimeFgap 39.9879
avgIndivPercentTimeBgap NaN
```

meanNumBgapsInMultTrackTraj NaN
growth_speed_mean_IncludeAllPause 39.4628
growth_speed_median_IncludeAllPause 30.3299
growth_lifetime_mean_IncludeAllPause 3.08163
growth_lifetime_median_IncludeAllPause 2
dynamicity 27.3094
avgComLatSec -0.21264
numNucleationEvents 518
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN

Dati LOG-Wiener

medNNdistWithinFrameMic 1.48437
nGrowths 521
growth_speed_median 30.7589
growth_speed_mean 40.0407
growth_speed_std 32.9122
growth_lifetime_median 2
growth_lifetime_mean 2.53071
growth_lifetime_std 1.4291
growth_length_median 1.23599
growth_length_mean 1.58556
growth_length_std 1.24883
nFgaps 36
fgap_speed_median 3.72357
fgap_speed_mean 3.98447
fgap_speed_std 11.9328
fgap_lifetime_median 4.5
fgap_lifetime_mean 4.40278
fgap_lifetime_std 1.74773
fgap_length_median 0.224133
fgap_length_mean 0.224437
fgap_length_std 0.876932
GrowthSpeedBeforeFgap_MicPerMin_mean 38.5784
GrowthSpeedBeforeFgap_MicPerMin_SE 4.11566
GrowthLifetimeBeforeFgap_Sec_mean 2.30769
GrowthLifetimeBeforeFgap_Sec_SE 0.15544
GrowthLengthBeforeFgap_Mic_mean 1.43394
GrowthLengthBeforeFgap_Mic_SE 0.171815
fgap_freq_time_mean 0.498616
fgap_freq_time_SE 0.0267842
fgap_freq_length_mean 1.02673
fgap_freq_length_SE 0.100401
nBgaps 4
bgap_speed_median -32.8572
bgap_speed_mean -34.8849

```
bgap_speed_std 7.75131
bgap_lifetime_median 1.5
bgap_lifetime_mean 1.5
bgap_lifetime_std 0.408248
bgap_length_median -0.956985
bgap_length_mean -0.88184
bgap_length_std 0.31054
GrowthSpeedBeforeBgap_MicPerMin_mean 42.1445
GrowthSpeedBeforeBgap_MicPerMin_SE 11.3868
GrowthLifetimeBeforeBgap_Sec_mean 2.875
GrowthLifetimeBeforeBgap_Sec_SE 0.688446
GrowthLengthBeforeBgap_Mic_mean 2.17102
GrowthLengthBeforeBgap_Mic_SE 0.694483
bgap_freq_time_mean 0.418651
bgap_freq_time_SE 0.101811
bgap_freq_length_mean 1.29308
bgap_freq_length_SE 0.923389
nGrowthTermEvents 463
GrowthSpeedBeforeTermEvent_MicPerMin_mean 40.4057
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.57756
GrowthLifetimeBeforeTermEvent_Sec_mean 2.5486
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0684814
GrowthLengthBeforeTermEvent_Mic_mean 1.60264
GrowthLengthBeforeTermEvent_Mic_SE 0.0592408
term_freq_time_mean 0.47922
term_freq_time_SE 0.00801889
term_freq_length_mean 1.05586
term_freq_length_SE 0.0381352
ratio_preFgapVel2preTermVel 0.954775
ratio_preBgapVel2preTermVel 1.04303
ratio_preFgapVel2preBgapVel 0.915383
ratio_preFgapLife2preTermLife 0.905476
ratio_preBgapLife2preTermLife 1.12807
ratio_preFgapLife2preBgapLife 0.802676
ratio_preFgapDisp2preTermDisp 1.35465
ratio_preFgapDisp2preBgapDisp 0.660491
percentTimeGrowth 88.9076
percentTimeFgap 10.6878
percentTimeBgap 0.404585
percentGapsForward 90
percentGapsBackward 10
percentGrowthLinkedBackward 0.767754
percentGrowthLinkedForward 7.4856
percentGrowthTerminal 88.8676
percentGrowthLinkedUndefinedGap 2.87908
ratio_TotalTracks2NumGrowthSubtracks 0.896353
NumTracksWithfGap2TotalTracks_Per 7.49465
NumTracksWithbGap2TotalTracks_Per 0.856531
NumTracksWithGap2TotalTracks_Per 7.92291
```

```
ratio_Compound2SingleTracks 0.1
VelGrowthInCompTrack_mean_MicPerMin 38.6388
VelGrowthInSingleTrack_mean_MicPerMin 40.4048
LifeGrowthInCompTrack_mean_Sec 2.75926
LifeGrowthInSingleTrack_mean_Sec 2.47561
DispGrowthInCompTrack_mean_Mic 1.7069
DispGrowthInSingleTrack_mean_Mic 1.55814
ratio_VelGrowthComp2Single 0.956292
ratio_LifeGrowthComp2Single 1.11458
ratio_DispGrowthComp2Single 1.09547
meanNumFgapsInMultTrackTraj 1.02857
avgIndivPercentTimeFgap 42.8883
meanNumBgapsInMultTrackTraj 1
avgIndivPercentTimeBgap 17.6282
growth_speed_mean_IncludeAllPause 39.2453
growth_speed_median_IncludeAllPause 30.0596
growth_lifetime_mean_IncludeAllPause 3.04536
growth_lifetime_median_IncludeAllPause 2
dynamicity 26.1181
avgComLatSec 0.336314
numNucleationEvents 465
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN
```

A.6 Statistiche di “Stack 27”

Dati originali

```
medNNdistWithinFrameMic 1.42996
nGrowths 458
growth_speed_median 32.2931
growth_speed_mean 39.9217
growth_speed_std 26.1834
growth_lifetime_median 2
growth_lifetime_mean 2.5393
growth_lifetime_std 1.49162
growth_length_median 1.22707
growth_length_mean 1.60181
growth_length_std 1.16445
nFgaps 39
fgap_speed_median -7.58224
fgap_speed_mean -5.98218
fgap_speed_std 15.5558
fgap_lifetime_median 3.5
fgap_lifetime_mean 3.44872
fgap_lifetime_std 1.66146
```

```
fgap_length_median -0.379112
fgap_length_mean -0.280331
fgap_length_std 0.764093
GrowthSpeedBeforeFgap_MicPerMin_mean 41.8359
GrowthSpeedBeforeFgap_MicPerMin_SE 4.45541
GrowthLifetimeBeforeFgap_Sec_mean 2.375
GrowthLifetimeBeforeFgap_Sec_SE 0.16187
GrowthLengthBeforeFgap_Mic_mean 1.62838
GrowthLengthBeforeFgap_Mic_SE 0.196621
fgap_freq_time_mean 0.483849
fgap_freq_time_SE 0.0256789
fgap_freq_length_mean 0.98837
fgap_freq_length_SE 0.127534
nBgaps 0
bgap_speed_median NaN
bgap_speed_mean NaN
bgap_speed_std NaN
bgap_lifetime_median NaN
bgap_lifetime_mean NaN
bgap_lifetime_std NaN
bgap_length_median NaN
bgap_length_mean NaN
bgap_length_std NaN
bgap_freq_time_mean NaN
bgap_freq_length_mean NaN
bgap_freq_time_SE NaN
bgap_freq_length_SE NaN
GrowthSpeedBeforeBgap_MicPerMin_mean NaN
GrowthSpeedBeforeBgap_MicPerMin_SE NaN
GrowthLifetimeBeforeBgap_Sec_mean NaN
GrowthLifetimeBeforeBgap_Sec_SE NaN
GrowthLengthBeforeBgap_Mic_mean NaN
GrowthLengthBeforeBgap_Mic_SE NaN
nGrowthTermEvents 402
GrowthSpeedBeforeTermEvent_MicPerMin_mean 39.9036
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.28161
GrowthLifetimeBeforeTermEvent_Sec_mean 2.56841
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0772856
GrowthLengthBeforeTermEvent_Mic_mean 1.6142
GrowthLengthBeforeTermEvent_Mic_SE 0.0581696
term_freq_time_mean 0.482644
term_freq_time_SE 0.00883111
term_freq_length_mean 0.995115
term_freq_length_SE 0.0374515
ratio_preFgapVel2preTermVel 1.04842
ratio_preBgapVel2preTermVel NaN
ratio_preFgapVel2preBgapVel NaN
ratio_preFgapLife2preTermLife 0.924697
ratio_preBgapLife2preTermLife NaN
```

```
ratio_preFgapLife2preBgapLife NaN
ratio_preFgapDisp2preTermDisp NaN
ratio_preFgapDisp2preBgapDisp NaN
percentTimeGrowth 89.6339
percentTimeFgap 10.3661
percentTimeBgap 0
percentGapsForward 100
percentGapsBackward 0
percentGrowthLinkedBackward NaN
percentGrowthLinkedForward 8.73362
percentGrowthTerminal 87.7729
percentGrowthLinkedUndefinedGap 3.49345
ratio_TotalTracks2NumGrowthSubtracks 0.884279
NumTracksWithfGap2TotalTracks_Per 8.88889
NumTracksWithbGap2TotalTracks_Per 0
NumTracksWithGap2TotalTracks_Per 8.88889
ratio_Compound2SingleTracks 0.104816
VelGrowthInCompTrack_mean_MicPerMin 39.3644
VelGrowthInSingleTrack_mean_MicPerMin 40.3476
LifeGrowthInCompTrack_mean_Sec 2.33553
LifeGrowthInSingleTrack_mean_Sec 2.56799
DispGrowthInCompTrack_mean_Mic 1.49859
DispGrowthInSingleTrack_mean_Mic 1.63444
ratio_VelGrowthComp2Single 0.975634
ratio_LifeGrowthComp2Single 0.909477
ratio_DispGrowthComp2Single 0.916882
meanNumFgapsInMultTrackTraj 1.08333
avgIndivPercentTimeFgap 40.2785
avgIndivPercentTimeBgap NaN
meanNumBgapsInMultTrackTraj NaN
growth_speed_mean_IncludeAllPause 38.7607
growth_speed_median_IncludeAllPause 31.2548
growth_lifetime_mean_IncludeAllPause 3.09666
growth_lifetime_median_IncludeAllPause 2
dynamicity 26.3383
avgComLatSec -0.421321
numNucleationEvents 405
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN
```

Dati Anscombe-Wiener

```
medNNdistWithinFrameMic 1.5166
nGrowths 448
growth_speed_median 32.3216
growth_speed_mean 39.1976
growth_speed_std 25.5421
```

```
growth_lifetime_median 2
growth_lifetime_mean 2.46094
growth_lifetime_std 1.44207
growth_length_median 1.21757
growth_length_mean 1.51733
growth_length_std 1.08235
nFgaps 35
fgap_speed_median -9.56932
fgap_speed_mean -9.10905
fgap_speed_std 14.9168
fgap_lifetime_median 3
fgap_lifetime_mean 3.34286
fgap_lifetime_std 1.88949
fgap_length_median -0.395528
fgap_length_mean -0.388128
fgap_length_std 0.769665
GrowthSpeedBeforeFgap_MicPerMin_mean 34.7116
GrowthSpeedBeforeFgap_MicPerMin_SE 3.00266
GrowthLifetimeBeforeFgap_Sec_mean 2.87143
GrowthLifetimeBeforeFgap_Sec_SE 0.313789
GrowthLengthBeforeFgap_Mic_mean 1.56037
GrowthLengthBeforeFgap_Mic_SE 0.180407
fgap_freq_time_mean 0.452379
fgap_freq_time_SE 0.0313871
fgap_freq_length_mean 0.945772
fgap_freq_length_SE 0.104389
nBgaps 5
bgap_speed_median -57.6403
bgap_speed_mean -59.0662
bgap_speed_std 14.7078
bgap_lifetime_median 2
bgap_lifetime_mean 1.9
bgap_lifetime_std 0.74162
bgap_length_median -2.01453
bgap_length_mean -1.78672
bgap_length_std 0.494439
GrowthSpeedBeforeBgap_MicPerMin_mean 38.9492
GrowthSpeedBeforeBgap_MicPerMin_SE 9.23754
GrowthLifetimeBeforeBgap_Sec_mean 3.2
GrowthLifetimeBeforeBgap_Sec_SE 1.00747
GrowthLengthBeforeBgap_Mic_mean 2.63607
GrowthLengthBeforeBgap_Mic_SE 1.41079
bgap_freq_time_mean 0.419048
bgap_freq_time_SE 0.0917825
bgap_freq_length_mean 1.09992
bgap_freq_length_SE 0.56406
nGrowthTermEvents 398
GrowthSpeedBeforeTermEvent_MicPerMin_mean 39.4512
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.32089
```

```
GrowthLifetimeBeforeTermEvent_Sec_mean 2.41457
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0699713
GrowthLengthBeforeTermEvent_Mic_mean 1.49247
GrowthLengthBeforeTermEvent_Mic_SE 0.0523235
term_freq_time_mean 0.495612
term_freq_time_SE 0.00833514
term_freq_length_mean 1.0263
term_freq_length_SE 0.0369834
ratio_preFgapVel2preTermVel 0.879862
ratio_preBgapVel2preTermVel 0.987275
ratio_preFgapVel2preBgapVel 0.891202
ratio_preFgapLife2preTermLife 1.18921
ratio_preBgapLife2preTermLife 1.32529
ratio_preFgapLife2preBgapLife 0.897321
ratio_preFgapDisp2preTermDisp 1.76624
ratio_preFgapDisp2preBgapDisp 0.591932
percentTimeGrowth 89.7071
percentTimeFgap 9.51993
percentTimeBgap 0.772986
percentGapsForward 87.5
percentGapsBackward 12.5
percentGrowthLinkedBackward 1.11607
percentGrowthLinkedForward 7.8125
percentGrowthTerminal 88.8393
percentGrowthLinkedUndefinedGap 2.23214
ratio_TotalTracks2NumGrowthSubtracks 0.890625
NumTracksWithfGap2TotalTracks_Per 7.26817
NumTracksWithbGap2TotalTracks_Per 1.25313
NumTracksWithGap2TotalTracks_Per 8.5213
ratio_Compound2SingleTracks 0.105114
VelGrowthInCompTrack_mean_MicPerMin 36.6609
VelGrowthInSingleTrack_mean_MicPerMin 39.5272
LifeGrowthInCompTrack_mean_Sec 2.88312
LifeGrowthInSingleTrack_mean_Sec 2.36648
DispGrowthInCompTrack_mean_Mic 1.67591
DispGrowthInSingleTrack_mean_Mic 1.47557
ratio_VelGrowthComp2Single 0.927486
ratio_LifeGrowthComp2Single 1.21832
ratio_DispGrowthComp2Single 1.13578
meanNumFgapsInMultTrackTraj 1.2069
avgIndivPercentTimeFgap 36.2517
meanNumBgapsInMultTrackTraj 1
avgIndivPercentTimeBgap 28.5641
growth_speed_mean_IncludeAllPause 38.2873
growth_speed_median_IncludeAllPause 31.7296
growth_lifetime_mean_IncludeAllPause 2.95278
growth_lifetime_median_IncludeAllPause 2
dynamicity 27.8626
avgComLatSec -0.59411
```

numNucleationEvents 396
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN

Dati LOG-Wiener

medNNdistWithinFrameMic 1.61168
nGrowths 390
growth_speed_median 30.1241
growth_speed_mean 37.5076
growth_speed_std 25.8956
growth_lifetime_median 2
growth_lifetime_mean 2.51795
growth_lifetime_std 1.5163
growth_length_median 1.16636
growth_length_mean 1.47632
growth_length_std 1.03538
nFgaps 27
fgap_speed_median -3.10292
fgap_speed_mean -2.37246
fgap_speed_std 12.633
fgap_lifetime_median 3
fgap_lifetime_mean 3.42593
fgap_lifetime_std 1.78511
fgap_length_median -0.250578
fgap_length_mean -0.128844
fgap_length_std 0.599466
GrowthSpeedBeforeFgap_MicPerMin_mean 35.7507
GrowthSpeedBeforeFgap_MicPerMin_SE 3.52798
GrowthLifetimeBeforeFgap_Sec_mean 2.2963
GrowthLifetimeBeforeFgap_Sec_SE 0.13413
GrowthLengthBeforeFgap_Mic_mean 1.37921
GrowthLengthBeforeFgap_Mic_SE 0.166918
fgap_freq_time_mean 0.473898
fgap_freq_time_SE 0.026302
fgap_freq_length_mean 0.955116
fgap_freq_length_SE 0.0903387
nBgaps 2
bgap_speed_median -44.4153
bgap_speed_mean -44.4153
bgap_speed_std 5.38504
bgap_lifetime_median 1.75
bgap_lifetime_mean 1.75
bgap_lifetime_std 0.353553
bgap_length_median -1.27958
bgap_length_mean -1.27958
bgap_length_std 0.104656

GrowthSpeedBeforeBgap_MicPerMin_mean 40.2949
GrowthSpeedBeforeBgap_MicPerMin_SE 12.4219
GrowthLifetimeBeforeBgap_Sec_mean 2.75
GrowthLifetimeBeforeBgap_Sec_SE 1.25
GrowthLengthBeforeBgap_Mic_mean 1.58806
GrowthLengthBeforeBgap_Mic_SE 0.270138
bgap_freq_time_mean 0.458333
bgap_freq_time_SE 0.208333
bgap_freq_length_mean 0.648464
bgap_freq_length_SE 0.110307
nGrowthTermEvents 352
GrowthSpeedBeforeTermEvent_MicPerMin_mean 37.8374
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.42197
GrowthLifetimeBeforeTermEvent_Sec_mean 2.52983
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0832331
GrowthLengthBeforeTermEvent_Mic_mean 1.48128
GrowthLengthBeforeTermEvent_Mic_SE 0.0549457
term_freq_time_mean 0.484317
term_freq_time_SE 0.00922069
term_freq_length_mean 1.04285
term_freq_length_SE 0.0394499
ratio_preFgapVel2preTermVel 0.94485
ratio_preBgapVel2preTermVel 1.06495
ratio_preFgapVel2preBgapVel 0.887226
ratio_preFgapLife2preTermLife 0.907688
ratio_preBgapLife2preTermLife 1.08703
ratio_preFgapLife2preBgapLife 0.835017
ratio_preFgapDisp2preTermDisp 1.07209
ratio_preFgapDisp2preBgapDisp 0.868488
percentTimeGrowth 91.0946
percentTimeFgap 8.58071
percentTimeBgap 0.324675
percentGapsForward 93.1034
percentGapsBackward 6.89655
percentGrowthLinkedBackward 0.512821
percentGrowthLinkedForward 6.92308
percentGrowthTerminal 90.2564
percentGrowthLinkedUndefinedGap 2.30769
ratio_TotalTracks2NumGrowthSubtracks 0.902564
NumTracksWithfGap2TotalTracks_Per 7.10227
NumTracksWithbGap2TotalTracks_Per 0.568182
NumTracksWithGap2TotalTracks_Per 7.67045
ratio_Compound2SingleTracks 0.0886076
VelGrowthInCompTrack_mean_MicPerMin 35.3821
VelGrowthInSingleTrack_mean_MicPerMin 38.3359
LifeGrowthInCompTrack_mean_Sec 2.54386
LifeGrowthInSingleTrack_mean_Sec 2.49051
DispGrowthInCompTrack_mean_Mic 1.43099
DispGrowthInSingleTrack_mean_Mic 1.48176

```
ratio_VelGrowthComp2Single 0.92295
ratio_LifeGrowthComp2Single 1.02142
ratio_DispGrowthComp2Single 0.965734
meanNumFgapsInMultTrackTraj 1.08
avgIndivPercentTimeFgap 38.1835
meanNumBgapsInMultTrackTraj 1
avgIndivPercentTimeBgap 26.0989
growth_speed_mean_IncludeAllPause 36.7851
growth_speed_median_IncludeAllPause 29.4826
growth_lifetime_mean_IncludeAllPause 2.96006
growth_lifetime_median_IncludeAllPause 2
dynamicity 23.9533
avgComLatSec -0.206108
numNucleationEvents 350
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN
```

A.7 Statistiche di “Stack 30”

Dati originali

```
medNNdistWithinFrameMic 1.52988
nGrowths 427
growth_speed_median 36.6645
growth_speed_mean 44.043
growth_speed_std 28.4599
growth_lifetime_median 2
growth_lifetime_mean 2.30562
growth_lifetime_std 1.1566
growth_length_median 1.35973
growth_length_mean 1.63192
growth_length_std 1.15283
nFgaps 35
fgap_speed_median -8.2803
fgap_speed_mean -5.52671
fgap_speed_std 20.4728
fgap_lifetime_median 4.5
fgap_lifetime_mean 3.72857
fgap_lifetime_std 1.6949
fgap_length_median -0.571852
fgap_length_mean -0.279866
fgap_length_std 0.992489
GrowthSpeedBeforeFgap_MicPerMin_mean 43.3949
GrowthSpeedBeforeFgap_MicPerMin_SE 3.89559
GrowthLifetimeBeforeFgap_Sec_mean 2.34722
GrowthLifetimeBeforeFgap_Sec_SE 0.189841
```

GrowthLengthBeforeFgap_Mic_mean 1.7739
GrowthLengthBeforeFgap_Mic_SE 0.239445
fgap_freq_time_mean 0.500178
fgap_freq_time_SE 0.028017
fgap_freq_length_mean 0.973365
fgap_freq_length_SE 0.123841
nBgaps 2
bgap_speed_median -71.6627
bgap_speed_mean -71.6627
bgap_speed_std 2.41246
bgap_lifetime_median 1.75
bgap_lifetime_mean 1.75
bgap_lifetime_std 0.353553
bgap_length_median -2.08306
bgap_length_mean -2.08306
bgap_length_std 0.351913
GrowthSpeedBeforeBgap_MicPerMin_mean 87.1306
GrowthSpeedBeforeBgap_MicPerMin_SE 48.841
GrowthLifetimeBeforeBgap_Sec_mean 2
GrowthLifetimeBeforeBgap_Sec_SE 0.5
GrowthLengthBeforeBgap_Mic_mean 2.49734
GrowthLengthBeforeBgap_Mic_SE 0.901944
bgap_freq_time_mean 0.533333
bgap_freq_time_SE 0.133333
bgap_freq_length_mean 0.46049
bgap_freq_length_SE 0.166311
nGrowthTermEvents 377
GrowthSpeedBeforeTermEvent_MicPerMin_mean 43.7838
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.48346
GrowthLifetimeBeforeTermEvent_Sec_mean 2.29443
GrowthLifetimeBeforeTermEvent_Sec_SE 0.06003
GrowthLengthBeforeTermEvent_Mic_mean 1.603
GrowthLengthBeforeTermEvent_Mic_SE 0.0580771
term_freq_time_mean 0.507407
term_freq_time_SE 0.00825542
term_freq_length_mean 0.991811
term_freq_length_SE 0.0385763
ratio_preFgapVel2preTermVel 0.991117
ratio_preBgapVel2preTermVel 1.99002
ratio_preFgapVel2preBgapVel 0.498045
ratio_preFgapLife2preTermLife 1.02301
ratio_preBgapLife2preTermLife 0.871676
ratio_preFgapLife2preBgapLife 1.17361
ratio_preFgapDisp2preTermDisp 1.55792
ratio_preFgapDisp2preBgapDisp 0.710316
percentTimeGrowth 88.0197
percentTimeFgap 11.6674
percentTimeBgap 0.312919
percentGapsForward 94.5946

percentGapsBackward 5.40541
percentGrowthLinkedBackward 0.468384
percentGrowthLinkedForward 8.43091
percentGrowthTerminal 88.2904
percentGrowthLinkedUndefinedGap 2.8103
ratio_TotalTracks2NumGrowthSubtracks 0.885246
NumTracksWithfGap2TotalTracks_Per 8.46561
NumTracksWithbGap2TotalTracks_Per 0.529101
NumTracksWithGap2TotalTracks_Per 8.73016
ratio_Compound2SingleTracks 0.10574
VelGrowthInCompTrack_mean_MicPerMin 43.4821
VelGrowthInSingleTrack_mean_MicPerMin 43.9711
LifeGrowthInCompTrack_mean_Sec 2.38194
LifeGrowthInSingleTrack_mean_Sec 2.29607
DispGrowthInCompTrack_mean_Mic 1.7514
DispGrowthInSingleTrack_mean_Mic 1.59779
ratio_VelGrowthComp2Single 0.988881
ratio_LifeGrowthComp2Single 1.0374
ratio_DispGrowthComp2Single 1.09614
meanNumFgapsInMultTrackTraj 1.09375
avgIndivPercentTimeFgap 42.9605
meanNumBgapsInMultTrackTraj 1
avgIndivPercentTimeBgap 20.8075
growth_speed_mean_IncludeAllPause 42.9269
growth_speed_median_IncludeAllPause 35.0449
growth_lifetime_mean_IncludeAllPause 2.84439
growth_lifetime_median_IncludeAllPause 2
dynamicity 31.9243
avgComLatSec -0.381262
numNucleationEvents 376
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN

Dati Anscombe-Wiener

medNNdistWithinFrameMic 1.62651
nGrowths 393
growth_speed_median 34.9159
growth_speed_mean 42.6115
growth_speed_std 27.4561
growth_lifetime_median 2
growth_lifetime_mean 2.38422
growth_lifetime_std 1.27023
growth_length_median 1.31526
growth_length_mean 1.64741
growth_length_std 1.22609
nFgaps 27
fgap_speed_median 10.4387

```
fgap_speed_mean 1.79972
fgap_speed_std 22.2595
fgap_lifetime_median 2.5
fgap_lifetime_mean 3.11111
fgap_lifetime_std 1.84669
fgap_length_median 0.307497
fgap_length_mean 0.0873784
fgap_length_std 0.99774
GrowthSpeedBeforeFgap_MicPerMin_mean 47.9827
GrowthSpeedBeforeFgap_MicPerMin_SE 4.46257
GrowthLifetimeBeforeFgap_Sec_mean 2.2963
GrowthLifetimeBeforeFgap_Sec_SE 0.220494
GrowthLengthBeforeFgap_Mic_mean 1.81728
GrowthLengthBeforeFgap_Mic_SE 0.24763
fgap_freq_time_mean 0.512728
fgap_freq_time_SE 0.0332242
fgap_freq_length_mean 0.794864
fgap_freq_length_SE 0.0904826
nBgaps 5
bgap_speed_median -59.1742
bgap_speed_mean -67.6406
bgap_speed_std 15.0064
bgap_lifetime_median 1
bgap_lifetime_mean 1.2
bgap_lifetime_std 0.447214
bgap_length_median -1.3552
bgap_length_mean -1.31178
bgap_length_std 0.371525
GrowthSpeedBeforeBgap_MicPerMin_mean 57.4451
GrowthSpeedBeforeBgap_MicPerMin_SE 4.16791
GrowthLifetimeBeforeBgap_Sec_mean 1.8
GrowthLifetimeBeforeBgap_Sec_SE 0.122474
GrowthLengthBeforeBgap_Mic_mean 1.71698
GrowthLengthBeforeBgap_Mic_SE 0.166126
bgap_freq_time_mean 0.566667
bgap_freq_time_SE 0.0408248
bgap_freq_length_mean 0.60074
bgap_freq_length_SE 0.0476293
nGrowthTermEvents 356
GrowthSpeedBeforeTermEvent_MicPerMin_mean 41.9029
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.47935
GrowthLifetimeBeforeTermEvent_Sec_mean 2.39747
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0682976
GrowthLengthBeforeTermEvent_Mic_mean 1.62779
GrowthLengthBeforeTermEvent_Mic_SE 0.0652566
term_freq_time_mean 0.497366
term_freq_time_SE 0.00889274
term_freq_length_mean 1.01234
term_freq_length_SE 0.040356
```

ratio_preFgapVel2preTermVel 1.14509
ratio_preBgapVel2preTermVel 1.37091
ratio_preFgapVel2preBgapVel 0.83528
ratio_preFgapLife2preTermLife 0.957799
ratio_preBgapLife2preTermLife 0.750791
ratio_preFgapLife2preBgapLife 1.27572
ratio_preFgapDisp2preTermDisp 1.05479
ratio_preFgapDisp2preBgapDisp 1.05842
percentTimeGrowth 91.2366
percentTimeFgap 8.17916
percentTimeBgap 0.584226
percentGapsForward 84.375
percentGapsBackward 15.625
percentGrowthLinkedBackward 1.27226
percentGrowthLinkedForward 6.87023
percentGrowthTerminal 90.5852
percentGrowthLinkedUndefinedGap 1.27226
ratio_TotalTracks2NumGrowthSubtracks 0.905852
NumTracksWithfGap2TotalTracks_Per 7.02247
NumTracksWithbGap2TotalTracks_Per 1.40449
NumTracksWithGap2TotalTracks_Per 8.42697
ratio_Compound2SingleTracks 0.0934579
VelGrowthInCompTrack_mean_MicPerMin 44.1132
VelGrowthInSingleTrack_mean_MicPerMin 42.0349
LifeGrowthInCompTrack_mean_Sec 2.37097
LifeGrowthInSingleTrack_mean_Sec 2.40031
DispGrowthInCompTrack_mean_Mic 1.7311
DispGrowthInSingleTrack_mean_Mic 1.62746
ratio_VelGrowthComp2Single 1.04944
ratio_LifeGrowthComp2Single 0.987775
ratio_DispGrowthComp2Single 1.06368
meanNumFgapsInMultTrackTraj 1.08
avgIndivPercentTimeFgap 37.3661
meanNumBgapsInMultTrackTraj 1
avgIndivPercentTimeBgap 20.6522
growth_speed_mean_IncludeAllPause 41.6767
growth_speed_median_IncludeAllPause 34.3636
growth_lifetime_mean_IncludeAllPause 2.78962
growth_lifetime_median_IncludeAllPause 2
dynamicity 33.796
avgComLatSec 0.123035
numNucleationEvents 356
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN

Dati LOG-Wiener

medNNdistWithinFrameMic 1.70386

```
nGrowths 361
growth_speed_median 36.0725
growth_speed_mean 43.2742
growth_speed_std 26.3189
growth_lifetime_median 2
growth_lifetime_mean 2.43352
growth_lifetime_std 1.5718
growth_length_median 1.2827
growth_length_mean 1.68097
growth_length_std 1.2432
nFgaps 19
fgap_speed_median 4.50433
fgap_speed_mean -0.456755
fgap_speed_std 21.116
fgap_lifetime_median 2.5
fgap_lifetime_mean 3.34211
fgap_lifetime_std 1.77992
fgap_length_median 0.14471
fgap_length_mean 0.0287623
fgap_length_std 1.01826
GrowthSpeedBeforeFgap_MicPerMin_mean 49.4523
GrowthSpeedBeforeFgap_MicPerMin_SE 6.33046
GrowthLifetimeBeforeFgap_Sec_mean 2.09524
GrowthLifetimeBeforeFgap_Sec_SE 0.156673
GrowthLengthBeforeFgap_Mic_mean 1.68143
GrowthLengthBeforeFgap_Mic_SE 0.248123
fgap_freq_time_mean 0.524603
fgap_freq_time_SE 0.0327654
fgap_freq_length_mean 0.856491
fgap_freq_length_SE 0.115653
nBgaps 3
bgap_speed_median -62.8521
bgap_speed_mean -66.3734
bgap_speed_std 15.7581
bgap_lifetime_median 1.5
bgap_lifetime_mean 1.33333
bgap_lifetime_std 0.288675
bgap_length_median -1.31685
bgap_length_mean -1.48475
bgap_length_std 0.541065
GrowthSpeedBeforeBgap_MicPerMin_mean 62.5849
GrowthSpeedBeforeBgap_MicPerMin_SE 21.2316
GrowthLifetimeBeforeBgap_Sec_mean 2.33333
GrowthLifetimeBeforeBgap_Sec_SE 0.440959
GrowthLengthBeforeBgap_Mic_mean 2.70715
GrowthLengthBeforeBgap_Mic_SE 1.29707
bgap_freq_time_mean 0.466667
bgap_freq_time_SE 0.101835
bgap_freq_length_mean 0.616052
```

```
bgap_freq_length_SE 0.290084
nGrowthTermEvents 329
GrowthSpeedBeforeTermEvent_MicPerMin_mean 42.7426
GrowthSpeedBeforeTermEvent_MicPerMin_SE 1.44537
GrowthLifetimeBeforeTermEvent_Sec_mean 2.45441
GrowthLifetimeBeforeTermEvent_Sec_SE 0.0897798
GrowthLengthBeforeTermEvent_Mic_mean 1.66823
GrowthLengthBeforeTermEvent_Mic_SE 0.0684779
term_freq_time_mean 0.500483
term_freq_time_SE 0.00946516
term_freq_length_mean 0.952722
term_freq_length_SE 0.038345
ratio_preFgapVel2preTermVel 1.15698
ratio_preBgapVel2preTermVel 1.46423
ratio_preFgapVel2preBgapVel 0.790163
ratio_preFgapLife2preTermLife 0.853664
ratio_preBgapLife2preTermLife 0.950671
ratio_preFgapLife2preBgapLife 0.897959
ratio_preFgapDisp2preTermDisp 1.62276
ratio_preFgapDisp2preBgapDisp 0.621109
percentTimeGrowth 92.8647
percentTimeFgap 6.71247
percentTimeBgap 0.422833
percentGapsForward 86.3636
percentGapsBackward 13.6364
percentGrowthLinkedBackward 0.831025
percentGrowthLinkedForward 5.81717
percentGrowthTerminal 91.1357
percentGrowthLinkedUndefinedGap 2.21607
ratio_TotalTracks2NumGrowthSubtracks 0.916898
NumTracksWithfGap2TotalTracks_Per 5.13595
NumTracksWithbGap2TotalTracks_Per 0.906344
NumTracksWithGap2TotalTracks_Per 6.0423
ratio_Compound2SingleTracks 0.0766667
VelGrowthInCompTrack_mean_MicPerMin 45.1085
VelGrowthInSingleTrack_mean_MicPerMin 43.0807
LifeGrowthInCompTrack_mean_Sec 2.38889
LifeGrowthInSingleTrack_mean_Sec 2.46333
DispGrowthInCompTrack_mean_Mic 1.79213
DispGrowthInSingleTrack_mean_Mic 1.68125
ratio_VelGrowthComp2Single 1.04707
ratio_LifeGrowthComp2Single 0.969779
ratio_DispGrowthComp2Single 1.06595
meanNumFgapsInMultTrackTraj 1.11765
avgIndivPercentTimeFgap 39.1365
meanNumBgapsInMultTrackTraj 1
avgIndivPercentTimeBgap 22.4349
growth_speed_mean_IncludeAllPause 42.3947
growth_speed_median_IncludeAllPause 35.2877
```

```
growth_lifetime_mean_IncludeAllPause 2.75439
growth_lifetime_median_IncludeAllPause 2
dynamicity 33.5316
avgComLatSec 0.0398791
numNucleationEvents 330
growth_density NaN
nucleationDensity NaN
fgap_density NaN
bgap_density NaN
```

Appendice B

Listati delle funzioni MATLAB

B.1 Calcolo rapporto segnale-rumore

```
function [peak_average,SNR]= SNRcount(nomefile)
%
%
% QUESTA FUNZIONE CALCOLA SNR E PSNR MEDIE DELLO STACK
% CONTENUTO IN 'NOMEFILE'
%
%
%
A=imread_tif(nomefile); %INTERO STACK
B=double(A);
dimension=size(B); %NORMALMENTE [256 256 120]
dim_frame=[dimension(1) dimension(2)]; % NORMALMENTE [256 256]

for i=1:dimension(3) %NORMALMENTE 120
    image_to_evaluate=B(:,:,i); %i-ESIMO FRAME
    % CALCOLO PSNR i-ESIMO FRAME
    background=image_to_evaluate(:,1:100);
    image_reference= imresize(background,dim_frame, 'nearest');
    peak_snr(i) = psnr(image_reference,image_to_evaluate,255);

    % CALCOLO DI SNR IN BASE A METRICA USUALE IN BIOLOGIA
    %(I0-IB)/IB

 sizeB=size(background);
    pot_back(i)= sum(sum(background.^2))/sizeB(1)/sizeB(2);
    SNR_bio_vector(i)=10*log10((255-sqrt(pot_back(i)))/sqrt(pot_back(i)));

end

% VALUTAZIONE DEI VALORI MEDI  SULL'INTERO STACK
```

```
peak_average=mean(peak_snr);
SNR=mean(SNR_bio_vector);
```

B.2 Analisi spettrale

```
function spectrum(nomefile)
%
close all
A=imread_tif(nomefile);
B=double(A);
B1=B(:,:,120);
FB1=fft2(B1);
MOD=abs(FB1);

massimo=max(max(MOD));
MODnorm=MOD./massimo;

MODshift=fftshift(MODnorm);
MODshiftdb=10*log10(MODshift);

[X Y]=meshgrid(-pi:pi/128:pi-pi/128, -pi:pi/128:pi-pi/128);

mesh(X,Y,MODshiftdb)
figure
unoDshift=MODshift(:,128);

semilogy(unoDshift)
grid

end
```

B.3 Conteggio numero di comete

```
function [n_comete,maxn,minn,med]=comet_count(movieInfo,n_frame)
%
% CONTEGGIO N. COMETE MINIMO, MASSIMO, MEDIO DELLO STACK
% GRAFICO ANDAMENTO N. COMETE CON IL TEMPO
%
close all
%
%movieInfo CONTIENE I RISULTATI ANALITICI IN USCITA DALL'ALGORITMO WATERSHED
%
for i=1:n_frame
    n_comete(i)= length(movieInfo(i).xCoord(:,1)); %NUMERO COMETE RILEVATE IN FRAME i
```

```
end;
maxn=max(n_comete);
minn=min(n_comete);
med=sum(n_comete)/n_frame;
figure
plot(n_comete)
grid
hold on
for i=1:n_frame
vetmedia(i)=med;
end;
plot(vetmedia,'--r');
%hold off
end
```

B.4 Elaborazione Anscombe-Wiener

```
function[B,noise]=anscombe_processing(nomefile)
%
%
% QUESTA FUNZIONE LEGGE LO STACK NOMEFILE.TIF
% ESEGUE LA TRASFORMATA DI ANSCOMBE
% ESEGUE FILTRAGGIO DI WIENER
% ESEGUE ANSCOMBE INVERSA
% SALVA L'INTERO STACK IN TEST.TIF (DA RINOMINARE OPPORTUNAMENTE)
% RESTITUISCE LO STACK ANCHE NELLA VARIABILE B
% RESTITUISCE LA VARIANZA DI RUMORE (CONSIDERATO GAUSSIANO BIANCO)
%
%

A=imread_tif(nomefile); %INTERO STACK
% B=double(A);
dimension=size(A);

AN=anscombe(A);
for t=1:dimension(3)
    [J(:,:,t),noise] = wiener2(AN(:,:,t),[3 3]);
end
BN=ianscombe(J);
B=uint8(BN);
imwrite_tif(B, 'test.tif');
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [ANSC] = anscombe(A)
```

```
% CALCOLA LA TRASFORMATA DI ANSCOMBE  DELLA MATRICE
%TRIDIMENSIONALE ANSC
B=double(A);
dimension=size(A); %NORMALMENTE [256 256 120]

for t=1:dimension(3)
    for i=1:dimension(1)
        for j=1:dimension(2)
            ANSC(i,j,t)=sqrt(B(i,j,t)+3/8);
        end
    end
end
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [B] =ianscombe(ansc)

% CALCOLA LA TRASFORMATA DI ANSCOMBE INVERSA DELLA MATRICE
%TRIDIMENSIONALE ANSC

dimension=size(ansc); %NORMALMENTE [256 256 120]

for t=1:dimension(3)
    for i=1:dimension(1)
        for j=1:dimension(2)
            B(i,j,t)=1/4*ansc(i,j,t)^2 -3/8;
        end
    end
end
end
```

B.5 Deconvoluzione Lucy-Richardson

```
function[B]=deconv_lr__processing(nomefile)
%
%
% QUESTA FUNZIONE LEGGE LO STACK NOMEFILE.TIF
% ESEGUE DECONVOLUZIONE LUCY-RICHARDSON
% SALVA L'INTERO STACK IN TEST.TIF (DA RINOMINARE OPPORTUNAMENTE)
% RESTITUISCE LO STACK ANCHE NELLA VARIABILE B
%
%
A=imread_tif(nomefile); %INTERO STACK
```

```
% B=double(A);
dimension=size(A);

BN = deconv_lucy_richardson(A);
B=uint8(BN);
imwrite_tif(B, 'test.tif');
end
%
%
function [B]=deconv_lucy_richardson(A)
dimension=size(A);
% CREAZIONE DI UN FILTRO BIDIMENSIONALE SIMMETRICO PASSABASSO GAUSSIANO
% DIMENSIONE[3,3], DEVIAZIONE STANDARD 0.5
% PER CAMBIARE USARE LA SINTASSI:
% h = fspecial('gaussian', hsize, sigma)

PSF = fspecial('gaussian');

% DECONVOLUZIONE CON METODO LUCY-RICHARDSON
% DI DEFAULT: 10 ITERAZIONI
% PER CAMBIARE USARE LA SINTASSI:
% J = deconvlucy(I, PSF, NUMIT)

for i=1:dimension(3)
    B(:,:,i) = deconvlucy(A(:,:,i),PSF);
end

end
```

Bibliografia

[1] M. Gai et al., in "ASPM and CITK regulate spindle orientation by affecting the dynamics of astral microtubules" 2016 (to be published).

[2] E. Meijering, I. Smal, G. Danuser, "Tracking in molecular bioimaging" in *IEEE Signal Processing Magazine*, v. 1, pp. 46–53, May 2006.

[3] I. Smal, M. Loog, W. Niessen, E. Meijering, "Quantitative comparison of spot detection methods in fluorescence microscopy" in *IEEE Transactions on Medical Imaging*, v. 29, no. 2, pp. 282–301, February 2010.

[4] N. Chenouard, I. Blochand, J. C. Olivo-Marin, "Multiple hypothesis tracking for cluttered biological image sequences" in *IEEE Transactions on Pattern Analysis and Machine Intell.*, v. 35, no. 11, pp. 2736–2750, November 2013.

[5] C. Kervrann, C. O. Sanchez-Sorzano, S. T. Acton, J. C. Olivo-Marin, M. Unser, "A guided tour of selected image processing and analysis methods for fluorescence and electron microscopy" in *IEEE Journal on Selected Topics in Signal Processing*, v. 10, no. 1, pp. 6–30, February 2016.

[6] B. Alberts, *Molecular biology of the cell* 6th edition, Garland Science, 2014.

[7] B. G. Katzung, *Farmacologia generale e clinica* 9th edition, Piccin-Nuova Libraria, 2014.

[8] I. Semenova, V. Rodionov, "Fluorescence microscopy of microtubules in cultured cells" in *Zhou, J. (ed.), Microtubule Protocols, Humana Press*, v. 137, pp. 93–102, 2007.

[9] K. T. Applegate, S. Besson, A. Matov, M. H. Bagonis, K. Jaqamal, G. Danuser, "plusTipTracker: quantitative image analysis software for the measurement of microtubule dynamics" in *Elsevier Journal of Structural Biology*, v. 176 (2011), pp. 168–184, 2011.

[10] A. V. Oppenheim, R. W. Schafer, *Digital signal processing* Prentice-Hall, 1975.

[11] F. Anscombe, "The transformation of Poisson, binomial and negative-binomial data" in *Biometrika*, v. 15, pp. 246–254, 1948.

[12] H. L. V. Trees, *Detection, estimation and modulation theory, Part III: radar-sonar signal processing and Gaussian signals in noise* John Wiley and sons, 2001.

[13] B. Zhang, J. Fadili, J. Starck, J. C. Olivo-Marin, "Multiscale variance-stabilizing transform for mixed Posson-Gaussian processes and its application in bioimaging" in *Proc. IEEE Intl. Conf. in Image Processing (ICIP)*, San Antonio, TX, USA, 2007, pp. 233–236.

[14] A. Bindilatti, N. Mascarenhas, "Patch reprojections for Non-Local methods" in *Signal Processing Letters*, v. 20, no.11, 2013.

[15] J. Boulanger, C. Kervrann, P. Bouthemy, P. Elbau, J. B. Sibarita, J. Salamero, "Patch-based nonlocal functional for denoising fluorescence microscopy image sequences" in *IEEE Transactions on Medical Imaging*, v. 29, no. 2, pp. 1522–1539, February 2010.

[16] M. Makitalo, A. Foi, "Optimal inversion of the generalized Anscombe transformation for Poisson-Gaussian noise" in *IEEE Transactions on Image Processing*, v. 2, no. 1, pp. 91–103, January 2013.

[17] B. Zhang, J. Fadili, J. Starck, "Wavelets, ridgelets and curvelets for Poisson noise removal" in *IEEE Transactions on Image Processing*, v. 51, no. 1, pp. 1093–1108, 2008.

[18] S. Mallat, *A wavelet tour of signal processing* Academic Press, 2008.

[19] D. L. Donoho, "De-noising by soft-thresholding" in *IEEE Transactions on Information Theory*, v. 41, no.3, pp. 613–627, May 1995.

[20] F. Luisier, C. Vonesch, T. Blu, M. Unser, "Fast interscale wavelet denoising of Poisson-corrupted images" in *Signal Processing*, v. 90, no. 2, pp. 415–427, 2010.

[21] F. Luisier, T. Blu, M. Unser, "Image denoising in mixed Poisson-Gaussian noise" in *IEEE Transactions on Image Processing*, v. 20, no. 3, pp. 696–708, March 2011.

[22] Y. LeMontagner, E. D. Angelini, J. C. Olivo-Marin, "An unbiased risk estimator for image denoising in the presence of mixed Poisson-Gaussian noise" in *IEEE Transactions on Image Processing*, v. 23, no. 3, pp. 1255–1268, March 2014.

[23] A. Bijaoui, G. Jammal, "On the distribution of the wavelet coefficients for a Poisson noise" in *Signal Processing*, v. 81, pp. 1789–1800, 2001.

[24] R. D. Novak, R. G. Baraniuk, "Wavelet-domain filtering for photon imaging systems" in *IEEE Transactions on Image Processing*, v. 8, no. 5, pp. 666–678, May 1999.

[25] L. Lo Presti, F. Neri, *L'analisi dei segnali* Torino, CLUT, 1992.

[26] J. M. Sanches, J. C. Nascimento, J. S. Marques, "Medical image noise reduction using Sylvester-Lyapunov equation" in *IEEE Transactions on Image Processing*, v. 17, no. 9, pp. 1522–1539, September 2008.

[27] K. E. Timmermann, R. D. Nowak, "Multiscale modeling and estimation of Poisson processes with application to photon-limited imaging" in *IEEE Transactions on Information Theory*, v. 45, no.3, pp. 846–862, April 1999.

[28] R. D. Nowak, E. D. Kolaczyn, "A statistical multiscale framework for Poisson inverse problem" in *IEEE Transactions on Information Theory*, v. 46, no. 8, pp. 1811–1825, August 2000.

[29] R. G. Brown, P. Y. C. Hwang, *Introduction to random signals and applied Kalman filtering* 4th edition, John Wiley and sons, 2012.

[30] C. Vonesch, M. Unser, "A fast thresholded Landweber algorithm for wavelet-regularized multidimensional deconvolution" in *IEEE Transactions on Image Processing*, v. 17, no. 4, pp. 539–549, April 2008.

[31] F. Soulez, "A learn 2D, apply 3D method for 3D deconvolution microscopy" in *Proc. IEEE Intl. Symp. Biomedical Imaging*, Beijing, China, 2014, pp. 1075–1078.

[32] S. Lefkimmiatis, M. Unser, "Poisson image reconstruction with Hessian Schatten-norm regularization" in *IEEE Trans. on Image Processing*, v. 22.

[33] J. C. Olivo-Marin, "Extraction of spots in biological images usin multiscale product" in *Pattern Recognition*, v. 35, no. 9, pp. 1989–1996, 2002.

[34] T. Lindeberg, *Scale-space Theory in Computer Vision* Springer-Verlag, 2010.

[35] D. Sage, F. R. Neuman, F. Hediger, S. M. Gasser, M. Unser, "Automatic tracking of individual fluorescent particles: application to the study of chromosome dynamics" in *IEEE Transactions on Image Processing*, v. 14, no. 9, pp. 1372–1383, September 205.

[36] I. Smal, K. Draegenstein, N. Galjart, W. Niessen, E. Meijering, "Particle filtering for multiple object tracking in dynamic fluorescence microscopy images: application to microtubule growth analysis" in *IEEE Transactions on Medical Imaging*, v. 27, no. 6, pp. 789–804, June 2008.

[37] R. Gonzales, R. Woods, *Digital image processing* Prentice-Hall, 2007.

[38] M. S. Grewal, A. P. Andrews, *Kalman filtering: theory and practice using MATLAB* 4th ed., John Wiley and sons, 2015.

[39] N. Chenouard et al., "Objective comparison of particle tracking methods" in *Nature methods*, v. 11, no. 3, pp. 281–290, March 2014.

[40] K. Jaqaman, D. Loerke, M. Mettlen, K. Kuwata, S. Grinstein, S. L. Schmid, G. Danuser, "Robust single-particle tracking in live-cell time-lapse sequences" in *Nature Methods*, v. 5, pp. 695–702, 2008.

[41] K. T. Applegate, G. Danuser, "Using plusTipTracker software to measure and analyze microtubule dynamics for +TIP comets" in *Technical Report*, 2012, pp. 1–27.

[42] [Online]: www.openmicroscopy.org/site/support/previous/omero3/omero.server

[43] F. Sgrò et al., "Tissue-specific control of midbody microtubule stability by Citron kinase through modulation og TUBB3 phosphorylation" in *Cell Death Differ.*, v. 23 (5), pp. 801–813, May 2016.