Εθνικο Μετσοβιο Πολυτεχνειο

Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων

Τομεας Τεχνολογιας Πληροφορικης και Υπολογιστων

Automatic Parameter Learning in BioImage Analysis Algorithms

Diploma Thesis

Sotirios Piliouras

Supervisors : Ivo Sbalzarini, Professor TU Dresden
Georgios Stamou, Professor NTUA

Athens, November 2019

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Automatic Parameter Learning in BioImage Analysis Algorithms

Diploma Thesis

Σωτήριος Πήλιουρας

Supervisors : Ivo Sbalzarini, Professor TU Dresden
Γεώργιος Στάμου, Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 12η Νοεμβρίου 2019.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Γεώργιος Στάμου	Νικόλαος Παπασπύρου	Ανδρέας Γεώργιος Σταφυλοπάτης
Καθηγητής Ε.Μ.Π.	Καθηγήτής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Νοέμβριος 2019

(Υπογραφή)

.....

Σωτηριος Πηλιουρας
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Το αντικείμενο της διπλωματικής αυτής εργασίας είναι ο αυτόματος υπολογισμός παραμέτρων σε αλγορίθμους ανάλυσης βιοεικόνας. Ο κλασικός τρόπος χειροκίνητης ρύθμισης των παραμέτρων παρουσιάζει ορισμένους περιορισμούς σε θέματα χρόνου, αποδοτικότητας, χρησιμοποιούμενης μνήμης και απαιτούμενης εμπειρίας του χρήστη. Την ίδια στιγμή η συνεχής αύξηση του όγκου των δεδομένων και των εικόνων στα βιολογικά πειράματα, απαιτεί εύρωστες και ευέλικτες λύσεις στα ζητήματα επεξεργασίας και ανάλυσης βιοεικόνας. Λαμβάνοντας υπόψιν τα παραπάνω προβλήματα, προτείνουμε ως λύση τη χρήση της design centering προσέγγισης. Έτσι λοιπόν, αναπτύξαμε μία εναλλακτική μέθοδο στη ρύθμιση παραμέτρων και την υλοποιήσαμε ως plugin για την πλατφόρμα ανάλυσης εικόνας Fiji. Στη συνέχεια, δοκιμάσαμε την επίδοση του προγράμματος μας για διαφορετικά δεδομένα εικόνων και πραγματοποιήσαμε μια αξιολόγηση ευχρηστίας (usability test). Η επιλογή του design centering για την επίλυση του συγκεκριμένου προβλήματος είχε ενθαρρυντικά αποτελέσματα ενώ η ανατροφοδότηση των συμμετεχόντων από την αξιολογιση ευχρηστιας ήταν αρκετά θετική. Τέλος, συμπεράναμε ότι ο χρόνος λειτουργίας μπορεί σε ορισμένες περιπτώσεις να είναι ένας περιορισμός, κάτι που προτείνεται να διερευνηθεί περαιτέρω για διαφορετικούς αλγαρίθμους.

Λέξεις κλειδιά: Design Centering, Parameter Learning, Image Analysis, Bioimage analysis, Fiji, Usability Test, Genetic Algorithm, Machine Learning, Statistical Methods, Robustness, Human-Computer Interaction, Parameter Tuning, Segmentation

Abstract

The present thesis addresses the problem of automatic parameter tuning in bioimage analysis algorithms. The "traditional" manual way of parameter tuning comes with some limitations in matters of time, accuracy, memory and required user experience. At the same time, the continuous increase of image set data in biology calls for robust and generic image analysis solutions. Taking these into consideration, we propose the use of the design centering approach to face both problems. We also introduce a method to do that and implement it as an interactive segmentation plugin for the Fiji bioimage analysis platform. Finally, we perform benchmark analyses, tests on public image collections and a usability study to test our software. Design centering had promising results in giving robust solutions for different kinds of biological data variations, while the automatic tuning plugin was well-accepted by the participants of our usability test. However, the running time can be in some cases a limitation, that needs to be further tested for other bioimage analysis algorithms.

Keywords: Design Centering, Parameter Learning, Image Analysis, Bioimage analysis, Fiji, Usability Test, Genetic Algorithm, Machine Learning, Statistical Methods, Robustness, Human-Computer Interaction, Parameter Tuning, Segmentation

Aknowledgements

I would like to express my deep gratitude to Prof. Dr. Ivo Sbalzarini for accepting my request to do my Master thesis at TU Dresden, as a member at the Max Planck Institute of Molecular Cell Biology and Genetics, and for his guidance and enthusiastic encouragement throughout this project. I would also like to express my very great appreciation to Prof. Dr. George Stamou for being my supervisor from my home university and for advising and supporting me all these months. My grateful thanks are also extended to the members of the MOSAIC Group for always being available to help and for creating a very joyful and welcoming environment.

I owe my deepest gratitude to my parents, Irene and Panagiotis, and my brother Giannis, for their endless support and help throughout my years of study. Finally, I wish to thank my friends for their encouragement and support.

Εκτεταμένη Περίληψη

0.1 Θεωρητικό Υπόβαθρο

0.1.1 Ανάλυση Βιοεικόνας και Κατάτμηση

Η ανάλυση βιοεικόνας (**BioImage Analysis**) είναι απλά η εξαγωγή πληροφοριών από βιολογικές εικόνες [1]. Η έρευνα στη σύγχρονη βιολογία εξαρτάται σημαντικά από τις εικόνες και τις πληροφορίες που εξάγωνται απο αυτές για να εξηγήσει και να αναλύσει τους μηχανισμούς της ζωής. Για να επιτευχθεί αυτό, χρησιμοποιείται πληθώρα βιολογικών εργαλείων τα οποία βασίζονται σε υπολογιστικές μεθόδους. Η αυξανόμενη εμπιστοσύνη και το ενδιαφέρον των βιολόγων στη βιοπληροφορική ανοίγει τον δρόμο σε πρωτότυπη έρευνα και νέες τεχνολογικές λύσεις [2]. Η παρουσία της ανάλυσης βιοεικόνας γίνεται ακόμα πιο αναγκαία όταν μιλάμε για μεγάλης κλίμακας βιολογικά πειράματα που μπορούν να έχουν εκατοντάδες ή χιλιάδες εικόνες, και εξασφαλίζει ότι τα αποτελέσματα θα είναι ακριβή, αντικειμενικά και αναπαραγώγιμα [2].

Η κατάτμηση αποτελέι μιά κατηγορία μέθοδων ανάλυσης βιοεικόνας, και είναι η διαδικασία διαίρεσης μιας ψηφιακής εικόνας σε πολλά τμήματα [3]. Ειδικότερα, στα αποτελέσματα της, κάθε σημείο (**pixel**) στην εικόνα πρέπει να βρίσκεται σε μια περιοχή ενώ κάθε περιοχή καθορίζεται από ένα σύνολο ιδιοτήτων [4]. Η κατάτμηση χρησιμοποιείται συχνά για την ανίχνευση αντικειμένων ή ορίων σε εικόνες με σκοπό την εξαγωγή πληροφοριών από τα δεδομένα ή την απλούστευση τους για περαιτέρω ανάλυση. Στη βιολογία, η κατάτμηση χρησιμοποιείται για την ταξινόμηση και ομαδοποίηση των εικονοστοιχείων ως αντικείμενα βιολογικού ενδιαφέροντος ή υποβάθρου [5]. Αφού τα αντικείμενα έχουν ταξινομηθεί, είναι δυνατόν να υπολογιστούν διάφορα ποσοτικά μεγέθη όπως το σχήμα, το μέγεθος και η υφή, που μπορούν να επιτρέψουν την περαιτέρω ταξινόμηση και ανάλυση.

Η ποσοτική αξιολόγηση των αποτελεσμάτων μιας μεθόδου κατάτμησης σε μία εικόνα γίνεται συνήθως μέσω της σύγκρισης με κάποια **ground truths**, τα οποία μπορούν να έχουν τη μορφή περιγράμματος των αντικειμένων, αριθμού των αντικειμένων, **foreground/background** , και βιολογικών ετικετών. Μία μετρική που συχνά χρησιμοποιείται για να βρεθεί η ομοιότητα μεταξύ δύο μασκών κατάτμησης βασίζεται στο **Jaccard Index** [9].

0.1.2 Χειροκίνητη Ρύθμιση Παραμέτρων και Διακύμανση Δεδομένων

Οι αλγόριθμοι ανάλυσης βιοεικόνων καλύπτουν ένα πολύ ευρύ φάσμα διαφορετικών μεθόδων και τεχνικών. Ανάλογα με το είδος της εικόνας και τη ζητούμενη διεργασία πάνω σε αυτήν, η δυσκολία χρήσης των αλγορίθμων ποικίλλει σημαντικά. Η χειροκίνητη ρύθμιση παραμέτρων είναι μια κοινή διαδικασία στην ανάλυση των βιολογικών δεδομένων, καθώς οι περισσότεροι αλγόριθμοι φέρουν ρυθμιζόμενες από τον χρήστη παραμέτρους, και είναι ουσιαστικά η διαδικασία Δοκιμής-Σφάλματος. Για τους αρχάριους χρήστες αυτό γίνεται συστηματικά με την αυξομείωση των παραμέτρων, κάτι που προσεγγίζει τον αλγόριθμο **hill climbing** [9]. Από την άλλη, οι έμπειροι

χρήστες συνήθως γνωρίζουν τις τροποποιήσεις των παραμέτρων που θα οδηγήσουν σε καλύτερη απόδοση και έτσι, αν υποθέσουμε ότι η διαίσθησή τους είναι σωστή, η διαδικασία μοιάζει με τον αλγόριθμο deepest ascent hill climbing .

Η χειροκίνητη ρύθμιση των παραμέτρων μπορεί να γίνει δύσκολη και χρονοβόρα, ειδικά για τους πιο περίπλοκους αλγορίθμους ή τις δύσκολες περιπτώσεις εικόνων. Επιπλεόν, οι χρήστες αυτών των αλγορίθμων έχουν συνήθως βιολογικό υπόβαθρο και άρα δεν είναι εξοικειωμένοι με τη λειτουργία και τις λεπτομέρειες των χρησιμοποιούμενων υπολογιστικών μεθόδων. Παρά τις πολλαπλές προσπάθειες για πλατφόρμες ευέλικτων λογισμικών και out-of-the-box ανάλυσης εικόνων, η εμπειρία έχει δείξει ότι σημαντική ανθρώπινη προσπάθεια δαπανάται για την ρύθμιση και τη διόρθωση λαθών σε παραμέτρους [5]. Ένας ακόμα περιορισμός της χειροκίνητης ρύθμισης παραμέτρων αφορά τη μνήμη, καθώς οι χρήστες πρέπει να αποθηκεύουν όλα τα αποτελέσματα προκειμένου να συγκρίνουν τη τρέχουσα έξοδο με τις προηγούμενες [10].

Η βιολογική πληροφορία συνήθως προέρχεται από συλλογές εικόνων όπως από διαφορετικά δείγματα από ένα πείραμα, από διαφορετικές όψεις του ίδιου δείγματος, αλληλουχίες εικόνων στο χρόνο (βίντεο) ή z-stacks ως αποτέλεσμα 3D απεικόνισης. Σε ότι αφορά την ανάλυση αυτών των δεδομένων, ο μεγάλος όγκος και η ποικιλία των εικόνων πολλές φορές καθιστά πρακτικά αδύνατη την ρύθμιση των παραμέτρων για κάθε εικόνα. Μια λύση σε τέτοιες περιπτώσεις είναι να βρεθεί ένα σύνολο παραμέτρων που δίνει κάλα αποτελέσματα σε ένα στιγμιότυπο των εικόνων και να γενικευθεί στις υπόλοιπες. Παρόλα αυτά, εάν δεν ληφθουν υπόψιν οι διακυμάνσεις που μπορεί να υπάρχουν στο σύνολο των εικόνων, αυτή η τακτική μπορεί να οδηγήσει σε λανθασμένα αποτελέσματα.

Λαμβάνοντας υπόψιν τα παραπάνω, βλέπουμε ότι η χειροκίνητη ρύθμιση παραμέτρων παρουσιάζει αρκετούς περιορισμούς, κάτι που μας οδηγεί στο συμπέρασμα ότι υπάρχει η ανάγκη για εναλλακτικές μεθόδους ρύθμισης παραμέτρων. Την ίδια στιγμή, σημαντικό χαρακτηριστικό αυτών των νέων μεθόδων κρίνεται η προσαρμογή στις διακύμανσεις των βιολογικών δεδομένων. Ως λύση σε αυτά, προτείνουμε στη παρούσα εργασία τη χρήση του design centering προβλήματος που παρουσιάζεται στη συνέχεια.

0.1.3 Design Centering

Ο στόχος του design centering είναι να προσδιοριστούν οι παράμετροι σχεδιασμού ενός συστήματος ή ενός μοντέλου που εγγυώνται τη λειτουργία του σε δεδομένες προδιαγραφές και είναι ανθεκτικές σε τυχαίες μεταβολές. Σε αντίθεση με τη βελτιστοποίηση, στην οποία προσπαθούμε να βρούμε τις παραμέτρους που ανταποκρίνονται καλύτερα στις προδιαγραφές του συστήματός μας, στο design centering θέλουμε τις παραμέτρους που πληρούν τις προδιαγραφές συμπεριφοράς του συστήματός μας με μεγαλύτερη ευρωστία [15]. Το design centering πρόβλημα χρησιμοποιείται ευρέως στη μηχανική ηλεκτρονικών κυκλωμάτων [26] ενώ πρόσφατα παρουσιάστηκαν εφαρμογές και στο πεδίο των συνθετικών βιολογικών κυκλωμάτων [27].

Η εφικτή περιοχή (feasible region) ορίζεται ως το σύνολο των σημείων στον χώρο παραμέτρων για το οποίο το σύστημα ή το μοντέλο ικανοποιεί όλες τις απαιτούμενες προδιαγραφές. Η περιοχή αυτή είναι κοινό να θεωρείται κυρτή, αλλά στην πραγματικότητα αυτό δεν συμβαίνει πάντα [16]. Ο όγκος της εφικτής περιοχής εκφράζει τη συνολική ποσότητα εφικτών σχεδιών του συστήματος και μπορεί να χρησιμοποιηθεί για τη σύγκριση και επιλογή μεταξύ διαφορετικών σχεδίων [17].

Οι διαφορετικές προσεγγίσης υπολογισμού του design center ενός συστήματος χωρίζονται σε δύο κατηγορίες: στις γεωμετρικές και στις στατιστικές. Οι γεωμετρικές μέθοδοι χαρακτηρίζονται

από τη χρήση απλών γεωμετρικών σχημάτων για την προσέγγιση της περιοχής εντός της οποίας πληρούνται οι προδιαγραφές συμπεριφοράς και συνήθως το κέντρο αυτού του σχήματος επιλέγεται ως το **design center**. Σημαντικό χαρακτηριστικό των γεωμετρικών μεθόδων είναι ότι η εφικτή περιοχή θεωρείται φραγμένη και κυρτή. Από την άλλη, οι στατιστικές μέθοδοι βασίζονται κυρίως στη **Monte Carlo** δειγματοληψία του χώρου παραμέτρων, και στην αξιολόγηση των σημείων του χώρου ως εφικτά ή μη. Δεδομένου οτι οι μέθοδοι **Monte Carlo** γίνονται υπολογιστικά δαπανηρές σε μεγάλες διαστάσεις, είναι σημαντικό να χρησιμοποιούνται έξυπνες τακτικές δειγματοληψίας που εστιάζουν στις πλούσιες πληροφοριακά περιοχές του χώρου [15]. Ένα μέτρο της ευρωστίας ενός σχεδιασμού του συστήματος είναι η αναλογία εφικτών προς μη εφικτών σημείων [21].

Συγκρίνοντας τις δύο αυτές κατηγορίες μεθόδων για τον υπολογισμό του **design center** μπορούμε να δούμε ότι μια σημαντική διαφορά είναι η δυνατότητα προσέγγισης ενός μη κυρτού χώρου. Έτσι για παράδειγμα, η χρήση γεωμετρικών μεθόδων για την προσέγγιση μιας μη κυρτής εφικτής περιοχής μπορεί να αποδειχθεί ανεπαρκής [16]. Ένα άλλο ενδιαφέρον χαρακτηριστικό που παρατηρείται είναι η δυνατότητα εκτίμησης του όγκου της εφικτής περιοχής, κάτι που αποτελεί σημαντικό πρόβλημα σε τομείς όπως η μηχανική λογισμικού, τα γραφικά υπολογιστών, τα οικονομικά και η στατιστική [25].

0.2 Μεθοδολογία και Υλοποίηση

0.2.1 Αλγόριθμοι και Εργαλεία

Στη παρούσα εργασία παρουσιάζουμε μία μέθοδο αυτόματου υπολογισμού παραμέτρων σε αλγορίθμους ανάλυσης βιοεικόνας την οποία υλοποίησαμε ως **plugin** για την πλατφόρμα ανάλυσης εικόνας **Fiji**[28]. Η βασική ιδέα είναι να δώσουμε μια εναλλακτική λύση στη χειροκίνητη ρύθμιση παραμέτρων που θα προσφέρει, παράλληλα, εύρωστα αποτελέσματα έναντι των διακυμάνσεων των δεδομένων. Για να το πετύχουμε αυτό, προσεγγίζουμε τον υπολογισμό των παραμέτρων ως ένα **design centering** πρόβλημα.

Δεδομένου ότι το **design centering** πρόβλημα δεν έχει χρησιμοποιηθεί προηγουμένως στο πεδίο της ανάλυσης βιοεικόνας, δεν είχαμε κάποιο παράδειγμα αλγορίθμου που να έχει δοκιμαστεί στην ακαδημαϊκή βιβλιογραφία ούτε προηγούμενη γνώση σχετικά με την κυρτότητα των εφικτών περιοχών του χώρου παραμέτρων. Έτσι, χρειαζόμασταν μια στατιστική μη κυρτή μέθοδο που θα μπορούσε να προσεγγίσει το πρόβλημα με έναν γενικότερο τρόπο. Ταυτόχρονα, θέλαμε μια μέθοδο αποδοτική σε χρόνο και σε δειγματοληψία. Για αυτούς τους δύο λόγους επιλέξαμε να χρησιμοποιήσουμε τον αλγόριθμο **Lp-Adaptation** που περιγράφεται παρακάτω.

Η ιδέα του **Lp-Adaptation** αλγορίθμου βασίζεται στη δειγματοληψία του χώρου παραμέτρων με ομοιόρφη πυκνότητα σε **Lp-Balls**. Οι **Lp-Balls** προσαρμόζουν δυναμικά τη θέση, τον προσανατολισμό και τις διαστάσεις τους ανάλογα με το ιστορικό δειγματοληψίας, μια διαδικασία εμπνευσμένη από τη βιολογία. Ο αλγόριθμος χρειάζεται ως είσοδο ένα αρχικό σημείο που ανήκει στην εφικτή περιοχή και μια συνάρτηση **oracle** η οποία δέχεται ένα υποψήφιο σημείο του χώρου και επιστρέφει εάν αυτό βρίσκεται στην εφικτή περιοχή ή όχι.

Τα βήματα που ακολουθούνται από τον αλγόριθμο είναι τα εξής:

1. Αρχικοποίηση → παρέχεται το αρχικό εφικτό σημείο και η συνάρτηση **oracle**
2. Δειγματολειψία → ομοιόρφη πυκνότητα σε **Lp-Balls**

3. Αξιολόγηση → βασισμένη στη συνάρτηση oracle
4. Προσαρμογή → βασισμένη στη μέθοδο Gaussian Adaptation

Για τον υπολογισμό του αρχικού εφικτού σημείου χρησιμοποιούμε τον μη-γραμμικό μη-κυρτό εξελικτικό αλγόριθμο βελτιστοποίησης CMA-ES [31].

Η συνάρτηση oracle κατασκευάζεται κατά την εκτέλεση του προγράμματος και βασίζεται σε πληροφορίες που δίνει ο χρήστης για την εικόνα. Η εξαγωγή αυτών των πληροφοριών από τον χρήστη γίνεται μεσω διάφορων queries και με τη βοήθεια των εργαλείων που προσφέρει το περιβάλλον της πλατφόρμας ανάλυσης εικόνας Fiji [28].

Για αυτή τη πρώτη απόπειρα αυτόματου υπολογισμού των παραμέτρων αποφασίσαμε να επικεντρωθούμε στο πρόβλημα της κατάτμησης και χρησιμοποιήσαμε συγκεκριμένα τον αλγόριθμο κατάτμησης Squassh [33].

0.2.2 Apeba Segmentation plugin

Το plugin που αναπτύξαμε για την πλατφόρμα Fiji ονομάστηκε Apeba (Automatic Parameter Estimation in BioImage Analysis). Η διαδικασία που ακολουθείται κατα την εκτέλεση του προγράμματος περιλαμβάνει τα εξής βήματα:

1. Εισαγωγή εικόνας: ο χρήστης επιλέγει την ακολουθία εικόνων προς ανάλυση και κατόπιν η ακολουθία ανοίγει ως παράθυρο ενσωματωμένο στο περιβάλλον της πλατφόρμας Fiji.
2. Επιλογή των key-frames: εδώ ο χρήστης επιλέγει τις εικόνες βάσει των οποίων γίνεται ο υπολογισμός του design center (περισσότερα στο κεφάλαιο 4.2.1).
3. Εισαγωγή της συνάρτησης oracle: σε αυτό το βήμα ο χρήστης δίνει διαφορετικές πληροφορίες σχετικά με την εικόνα, με τη βοήθεια της διεπαφής και των εργαλείων της (4.2.2-4.2.8).
4. Υπολογισμός του αρχικού εφικτού σημείου: χρησιμοποιείται η συνάρτηση oracle και ο αλγόριθμος CMA-ES για τον υπολογισμό του αρχικού εφικτού σημείου (4.2.9).
5. Υπολογισμός των design center: ο αλγόριθμος Lp-Adaptation χρησιμοποιείται για τον υπολογισμό του design center για κάθε επιλεγμένη εικόνα της ακολουθίας (4.2.10).
6. Τελική κατάτμηση: Ο αλγόριθμος Squassh εκτελείται με τις παραμέτρους να ακολουθούν τη γραμική παρεμβολή των design centers .

0.3 Αποτελέσματα

Οι δοκιμές που κάναμε στο πρόγραμμα μας και τα αποτελέσματα που πήραμε χωρίζονται σε τρείς κατηγορίες:

- Ευρωστία έναντι Διακύμανσης των Δεδομένων: δοκιμάσαμε την επίδοση του προγράμματος μας για διαφορετικά δεδομένα εικόνων και σε 4 διαφορειτκά είδη διακυμάνσεων (διαφορετικές όψεις του ίδιου δείγματος, αλληλουχίες εικόνων στο χρόνο, z-stacks, διαφορετικά stains). Ο κύριος σκοπός αυτών των δοκιμών ήταν να δούμε εάν όντως το design centering δίνει εύρωστα αποτελέσματα και καλύτερη απόδοση σε δεδομένα με διακυμάνσεις (περισσότερα στο κεφάλαιο 5.1).

- **Benchmarking**: όπου παρουσιάζουμε τα αποτελέσματα του αλγόριθμου **Lp-Adaptation** για διαφορετικό αριθμό επαναλήψεων και τους συνολικούς χρόνους εκτέλεσης (περισσότερα στο κεφάλαιο 5.2).
- Αξιολόγηση Ευχρηστίας: σε αυτή τη δοκιμή, ερευνητές από το Ινστιτούτο **Max Planck** χρησιμοποίησαν το πρόγραμμα μας σε 3 διαφορετικές περιπτώσεις εικόνων και μας έδωσαν το **feedback** τους κάνοντας τη σύγκριση με τη διαδικασία της χειροκίνητης ρύθμισης των παραμέτρων (περισσότερα στο κεφάλαιο 5.3).

Contents

1 Introduction

Manual parameter tuning is a common process in bioimage analysis algorithms that requires the analyst to do a repetitive task of parameter "tweaking", algorithm running and output inspection. This process can become difficult and time consuming, especially when working with complex algorithms or difficult images. Past efforts have been made to facilitate this process either by visualizing the parameter space or by using optimization algorithms to tune the parameters. However, one common limitation of these approaches is that they do not consider the variation that the modern biological data can have. As the volume of data from biological experiments increases rapidly, the latter is an important issue in today's bioinformatics that has created new needs for more robust and automatic solutions.

Taking these into account, in the present thesis we are introducing a new way of automatic tuning based on the design centering problem that aims to provide robust parameters for bioimage analysis algorithms. In design centering the objective is to determine the design parameters of a system or model that guarantee operation within given specifications and are robust against random variations. Design Centering is a long standing problem in electronic circuits engineering, where the design parameters frequently deviate from their nominal values due to the process variations, and has also found applications in the field of synthetic biological circuits. The various approaches to design centering are divided into two categories: geometrical and statistical. Another way to categorize these approaches would be to convex and non-convex ones.

To estimate the design centers we are using the Lp-Adaptation algorithm which is a non-convex statistical method, inspired by the evolution of robustness in biological systems and by randomized schemes for convex volume computation. The main steps of Lp-Adptation are: 1) Initialization, 2) Sampling, 3) Evaluation, 4) Adaptation. For the initialization step we need an oracle and a start point. An oracle is a function that takes the sampled points as input and returns if they are in the feasible region or not. The oracle in our case is based on user input about the image which is given through an interactive interface. In order to find a feasible point, we use the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) which is a non-convex black-box evolutionary optimization algorithm. All this is implemented as a plugin for the popular open-source image analysis framework Fiji.

To test the robustness of our software against the data variations we tuned the parameters for various public image sets that presented different kinds of variations. Then, we performed benchmark analyses for the time and performance of the plugin. Finally, we conducted a usability test with researchers as participants that are relevant to the field.

This manuscripts presents our contributions in the field of bioimage analysis, which are the following:

1. Designing an assisted interactive parameter tuning method that gives robust solutions for bioimage analysis algorithms, and implementing it as a Fiji plugin.
2. Testing the design centering approach for the parameter tuning task and comparing it with the manual and optimization ones.

The thesis is organized as follows:

- In Chapter 2, the background and preliminaries to our project are explained.
- In Chapter 3, the methodology and the different tools used are presented.
- In Chapter 4, the step by step process and the code structure of our software are analyzed.
- Chapter 5 consists of our tests and results including benchmark analyses, tests on public image collections and a usability study.
- Chapter 6 cocnludes this thesis and presents future outlook.

2 Background

2.1 BioImage Analysis

BioImage analysis is simply the extraction of information from biological pictures [1]. Research in modern biology depends significantly on image data and the information extracted from them in order to explain and analyze the mechanisms of life. To achieve that many new biological tools have relied on computational methods. It is considered that advances in the field of bioimage computation will not only be the foundation for new imaging methods but also the way to achieve new biological discoveries that otherwise would not be possible [2]. At the same time, the trust and interest of biologists in bioinformatics is increasing, leading to more possibilities for novel research and space for new informatics solutions.

The presence of bioimage analysis becomes essential when we are talking about large scale biological experiments that can have hundrends or thousands of images [2]. These data are often a result of automated microscopy and can be e.g. images from different samples, time series or z-stacks. Their big size and variation makes it hard to gain knowledge from them but image analysis can help in extracting information and ensure that the results will be accurate, objective and reproducible.

2.1.1 Image Segmentation

Image segmentation is the process of partitioning a digital image into multiple segments (or sets of pixels) [3]. More specifically in image segmentation every point (pixel) in the image must be in a region while every region is determined by a set of properties [4]. Segmentation is often used for detecting objects or boundaries in images and it helps in getting information about the data or making them simpler for further analysis. In bioimage analysis, segmentation is used to classify and group pixels as objects of biological interest or background [5]. After the objects have been segmented, it is possible to compute various quantitative descriptors such as dynamics, shape, size or texture which can then enable even further classification and analysis.

Segmentation Techniques

There are several techniques used for image segmentation:

- **Thresholding Methods** are the simplest methods in which a gray-scale image is turned into binary by applying a threshold value, resulting in a segmentation mask. There are various methods for the threshold selection that can be based, as described by Sezgin et al. [6], on the histogram of the image, the entropy, the spatial dependencies of the pixels and more.

- **Edge Based Methods** rely on the exploration of rapid changes in the intensities values between the pixels of the image, with the aim to detect the edges of the objects of interest. These methods can have good performance in images with high contrast [7].
- **Region Based Methods** function by partitioning the image into homogeneous regions. This can be done either by the growing of seeds which are initial pixels in the image, or by splitting and merging regions of the image in respect to some characteristic.
- **Clustering Methods** rely on the division of the image into homogeneous clusters. In soft clustering, which is a subcategory of these methods, the pixels can be classified in more than one cluster which makes it a more realistic approach.
- **Watershed methods** treat the image like a topographic map, with the brightness of each point representing its height, and find the lines that run along the tops of ridges. This methods are in general stable and they result in continuous boundaries.
- **Partial Differential Equation Based Methods** are fast segmentation methods that can be used e.g. to enhance the edges and remove the noise from the image.
- **Artificial Neural Network Based Segmentation Methods** are used widely in medical imaging projects but they often require more time than other methods because of their training step and training data.

Accuracy Assessment Methods

When working with image segmentation there is an important question to consider: "What is a good segmentation?". Having ways to assess the segmentation results is essential in developing, testing and publishing new image analysis algorithms for the life sciences. Also, having some points of reference in matters of performance is the most reliable way to compare already existing segmentation methods. Back to our question, according to Haralick et al. [8], a good segmentation is expected to have the following:

- Regions that are uniform and homogeneous with respect to some characteristic such as texture or brightness.
- Region interiors that are simple without many holes.
- Adjacent regions that have significantly different values in respect to the characteristic on which they are uniform.
- Boundaries that are spatially accurate and not ragged.

But when it comes to bioimage analysis, the segmentation tasks can be specific and a single image can have multiple kinds of segmentations that are considered good in different contexts and for different purposes. For example, in an image with connected cells we might want to segment each cell seperately but also to find the boundaries of the whole area covered by the bundle of cells. So we see that the segmentation in this case is largely task-related which makes it more difficult to find quantitative methods to assess its accuracy.

Traditionally, the assessment of bioimage segmentations is done by comparing them to synthetic data or to human-produced ground truths of real data. These ground truths can

be e.g. object counts, foreground/background, outlines of objects or biological labels. After that, it remains to determine the metrics that will be used for the comparison. When it comes to the foreground/background type, a common practice is to use the Jaccard index (or else known as Intersection Over Union) to compare the similarity between the ground truth and the segmentation mask [9]. If S_{gt} is the ground truth represented by a set of pixels and S_{res} the method's segmentation result, then the Jaccard similarity $O_j \in [0, 1]$ is defined as:

$$O_j = \frac{|S_{gt} \cap S_{res}|}{|S_{gt} \cup S_{res}|} \tag{2.1}$$

Of course, another way would be to directly ask experienced people in bioimage analysis to evaluate the segmentation results in relation to a specific task.

2.1.2 Manual Parameter Tuning, Limitations and Data Variation

BioImage analysis algorithms cover a very wide spectrum of different methods and techniques. Depending on the kind of image and the task, there are algorithms that can be from very simple and intuitive to much more complex or even black-box. Manual parameter tuning for these algorithms is a common process in the analysis of biological data. Most image analysis algorithms have user-specified parameters that affect their output and they can vary depending e.g. on the observed specimen type, the imaging technique or the image analysis task. The conventional approach for tuning these algorithms is to find their parameters by trial and error as shown in Figure 2.1. For non-expert users this is usually performed systematically by incrementing or decrementing the parameters, which resembles a component wise hill climbing algorithm [9]. On the other hand, experts in bioimage analysis often know the parameters modifications that will result in better segmentation performance so, if we assume that their intuition is right, then it resembles the steepest ascent hill climbing algorithm.

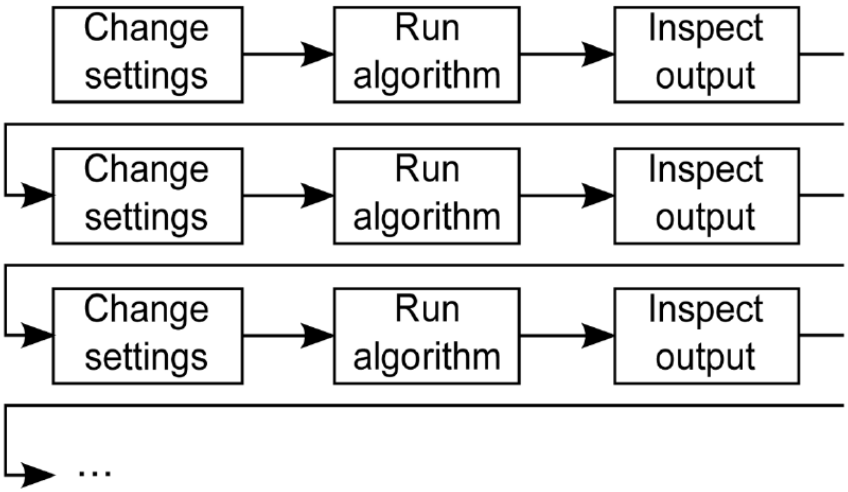


FIGURE 2.1: The conventional approach for parameter optimization. Based on a qualitative assessment of current output, users iteratively change parameter settings and run algorithms to generate new output. This process is repeated until the output quality is satisfactory. (borrowed by Pretorius et al. 2011 [10])

Manual tuning can become difficult and time consuming, especially for more complex algorithms or the "hard" image cases. It is important also to note that the users of these algorithms come mostly from biological or related backgrounds so the computational internal processes might not be familiar to them. What can make this process even more ambiguous is the fact that one cannot be always sure if the algorithm used can manage to perform the analysis task in a satisfying level. Which means that sometimes the amount of time spend for searching the "right" parameters turns out to be futile. Furthermore, despite the multiple efforts to flexible software platforms for building image analysis pipelines that work out of the box, practice has shown that substantial human effort is spent on tuning parameters and correcting faulty output [5]. Lastly, an issue that arises with manual tuning is the memory load as the users have to rely on memory recall to compare the current output with previous ones [10].

Biological data usually come in collections of images such as screenings of different samples from an experiment, images from different views of the same sample, time series (video) of biological processes or z-stacks as a result of 3D imaging. Analyzing these data often requires the tuning of parameters for bioimage analysis algorithms. The big volume and variety of images though, can make it practically impossible to tune the parameters for each single image. One solution is to find a set of parameters that will work for one instance of the image collection and use it for the rest of them. However, in this way the variation between the different images is not taken into account and so it may lead to erroneous in average analysis results.

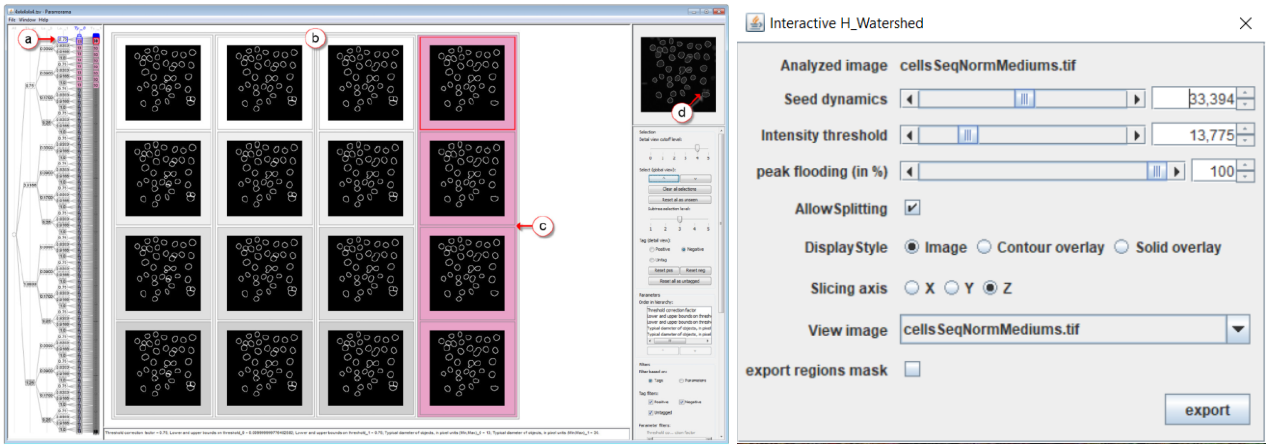
Taking into consideration the above we can conclude that manual tuning comes with some limitations that call for alternative approaches.

2.2 Alternative Approaches to Manual Parameter Tuning in Bioimage analysis algorithms

In this section we are going to review some previous bioimage analysis approaches that can be seen as alternatives to the manual tuning process.

One idea is the use of machine learning for automatic and interactive bioimage analysis tools. As Meijering et al. [5] state "we expect bioimage analysis solutions to become increasingly automated and robust". This approach not only could make the tuning process easier and faster but also more open to people with less experience. It is worth mentioning that despite the fact that non-expert image analysts tend to be intimidated by the concepts of machine learning, in reality these tools often require less expertise than tools that use manually configured algorithms [2]. One example in this category is Ilastik [11]. This open-source interactive learning and segmentation toolkit enables researchers to train a machine learning algorithm in order to identify which pixels of an image belong to which class of interest, based on labels provided by the user through a mouse interface. The learning step is performed by a random forest classifier in which each pixel's neighborhood is characterized by a set of generic features. Then the user can interactively alter the segmentation result and further tune the classifier, which in the end can be exported and used to automatically process large number of images. Ilastik is robust, with high segmentation performance in standard problems and has become a very common tool in the bioimage analysis community. We can find many examples in this category of automatic or semi-automatic image analysis like Cell Profiler [12] and Weka Segmentation [13]. However, we should note that these approaches automate the bioimage analysis process by introducing new techniques, instead of dealing with the automation of the already existing non-machine learning ones.

Focusing more on the effort to actually facilitate the manual tuning process of already existing algorithms we found another idea based on the visualization of the parameter space. In this case, the objective is to enable the users to explore the parameter space in a more structured and quick way. A first example of this approach is a plugin for CellProfiler and a visualization tool called Paramorama, proposed by Pretorius et al. [10]. In Paramorama the parameter optimization is based on parameter sampling and interactive visual exploration. At first users are choosing for each parameter a range of the values to explore and a number of samples. Then, after running the algorithm for all the parameter combinations, they can view the results in a hierarchical clustering tree and, finally, select the images that have good or bad results. One limitation of this method is that it does not offer an exhaustive search of the parameter space. We can see the visualization with Paramorama in Figure 2.2a. A second example of parameter visualization is the Interactive Watershed developed for ImageJ/Fiji [14] that explores the parameter space of the Watershed segmentation algorithm [Vincent 1991]. In initialization the plugin calculates the full watershed of the image for all local maxima and it generates a tree whose nodes correspond to segments in the image. After that, the user can change the values of the parameters by using the scroll-bars as shown in Figure 2.2b, and watch the segmentation results update instantly in another image window. This method works fast and is fully interactive, but it is only implemented for a specific algorithm, which also runs fast.



(A) (B)

FIGURE 2.2: (A) The Paramorama clustering tree visualization of the parameter space [10]. (B) The UI of the Interactive Watershed, where the user can control the scroll-bar to explore the different parameters and watch their results by using the three slide bars [14].

Another alternative approach to manual tuning is the idea to let optimization algorithms solve the parameter fitting problem. An example to this approach was introduced by Held et al. [9], who compared different optimization algorithms for automatic parameter fitting in a microscopy image segmentation. The algorithms used were: Hill Climbing, Steepest Ascent Hill Climbing, Coordinate Descent and Genetic Algorithm. The results indicated that the genetic algorithm outperforms other approaches in solving the optimization problem, but its convergence time was the highest. An interesting finding from the same study was that most parameter spaces of the used images contained several performance maxima, which can be an important challenge for optimization methods.

To sum up, we have seen that there are many approaches in automating the bioimage analysis process with promising results but they mainly focus on introducing new machine learning techniques. On the other hand, the approaches that actually aim to ameliorate the manual tuning process of existing methods may lack in exploring exhaustively the parameter space or in addressing more costly in time algorithms. Finally, an important common limitation for these methods is that they do not consider the data variation mentioned in the previous section, and so generalizing the parameters found by them for image collections is considered questionable.

2.3 Design Centering

2.3.1 Definitions

The objective of **design centering** is to determine the design parameters of a system or model that guarantee operation within given specifications and are robust against random variations. Opposed to Design Optimization, in which we try to find the parameters that best fulfill the specifications of our system, in Design Centering we want the parameters that fulfill the specifications of our system most robustly [15].

The **feasible region** is defined as the set of points in the design parameter space for which the system or model satisfies all specifications on its behavior. It is common to assume that this region is convex, but in reality this is not always the case [16]. Actually, most of the times we cannot guarantee the convexity of the feasible region and that makes it difficult to decide what design centering approach to use as we will see below.

The **volume** of the feasible region is a measure of the total amount of feasible designs available and can be used to compare and choose between different designs or competing models [17].

The design center [18] is described as the point among all parameter space that represents the mean of a probability distribution of maximal volume covering the feasible region with a given target hitting **probability P**.

2.3.2 Different Approaches

The approaches to design centering can be divided into two categories: geometrical and statistical.

Geometrical approaches are utilizing simple geometric bodies to approximate the region where the behavioral specifications are satisfied. Then, usually the center of this body is chosen as the design center. In these cases the feasible region is mostly assumed to be bounded and convex. Some examples of geometrical approaches are:

1. The Ellipsoidal Approximation [19] which finds the ellipsoid of largest volume that fits inside the feasible region while the center and all endpoints of the ellipsodial axes are feasible points. One drawback of this method is that an ellipsoid is symmetric to any hyperplane passing through its center and so it makes it difficult to approximate less symmetric areas [16].
2. The Simplical Approximation [20] which is based on the approximation of the feasible region's boundary by a convex polytope. While a polytope can provide a better approximation than an ellipse to a convex structure (since it can be seen as polytope with infinite faces) finding its center is complex and costly in time. In figure 2.3 we can see the approximation of a convex area by a polytope.
3. The Polytope Method [16] which uses a convex polytope to approximate the feasible region and then it finds the design center by either inscribing the largest Hessian ellipsoid or by using a convex programming approach.

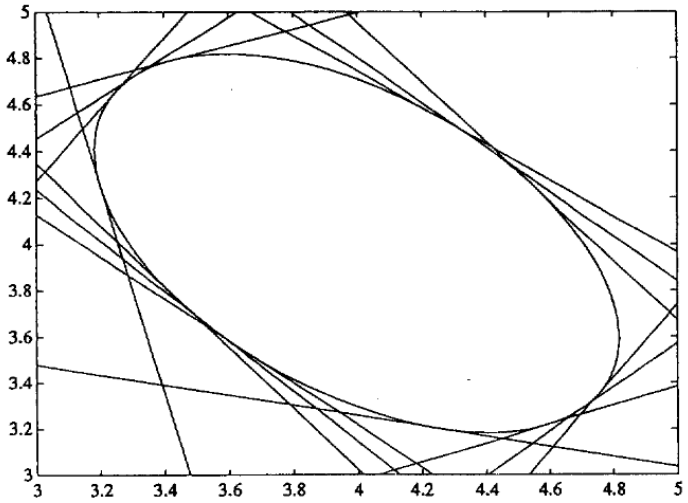


FIGURE 2.3: Approximating an ellipse by a polytope.[16]

The Statistical Approach mainly relies on Monte Carlo sampling of the parameter space. Since Monte Carlo methods can get computationally expensive in higher dimensions, it is important to find a smart sampling proposal and focus on the informative areas of the space[15]. Every sampled point is then evaluated and classified as feasible or not. A measure of robustness of the design is the ration of feasible to infeasible points [21]. Some examples of statistical approaches are:

1. The Constraint adaptation by Diferential Evolution [22] is a statistical method which assumes the feasible region to be convex. It starts from a population of initial points that loosely satisfy the specifications and then the specifications are tightened until reaching the original ones. Then the design center is estimated by finding the mean of the final points.

2. The Advanced First-Order Second Moment (AFOSM) method [23] in which points are samples from Lp-balls in order to estimate the ratio of feasible to infeasible points and approximate the feasible region. One drawback of this method is that it does not allow to estimate the total volume of the feasible region, as the Lp-ball must be completely inside the feasible region.

3. The Center of Gravity Method [24] which computes in each iteration the center of gravity for the feasible and the infeasible sampled points. Then the design center is gradually moving away from the infeasible points and towards the feasible ones.

Comparing the two design centering estimation approach types we can see that an important difference is the capability to approximate a non-convex space. Which means that the use of geometrical (or other convex) approaches for the approximation of a non-convex feasible region can be proven inadequate [16]. Another interesting characteristic to observe, is the capability of estimating the volume of the feasible region, which is an important problem in many areas including software engineering, computer graphics, economics, and statistics [25].

2.3.3 Applications

Design Centering has been mostly explored in electronic circuit engineering. Automatic design methods are an important need in keeping up with the ever growing demands on the designers' productivity and time-to-market [26]. In circuits manufacture, the design parameters frequently deviate from their nominal values due to manufacturing process variations [16]. This results in circuits that don't meet some behavioral specifications, such as requirements on the delay, gain, frequency response, harmonic distortion, energy consumption or manufacturing cost. In that case, design centering can be used to determine the nominal values of these parameters, which are mainly electronic components, such that the circuit fulfills some specifications and is robust against manufacturing tolerances in the components [15]. More recently, there were new applications in the field of synthetic biology where the aim is to robustly design novel synthetic biological circuits [27].

3 Methodology

In this project we are introducing an automatic process for parameter estimation in bioimage analysis algorithms called Apeba (Automatic Parameter Estimation in Bioimage Analysis). The main idea is to provide an alternative solution to the manual tuning problem that is also robust against the biological data variations. In order to do that we are approaching the parameter tuning task as a design centering problem. We implemented Apeba as a plugin for the free open-source bioimage processing framework Fiji, as part of the Mosaic Suite. In this chapter we are going to present the methods and tools used for Apeba and its development.

3.1 Lp-Adaptation Design Centering Method

Since the design centering problem has not been used before for bioimage analysis purposes, there was no example of a design centering method used for this task in the academic literature or any prior knowledge about the convexity of feasible regions in bioimage analysis algorithms. Therefore, we needed a statistical non-convex method that could approach the problem in a more general way. At the same time, image analysis in biology can be costly in time due to big data or complex algorithms, so we also wanted a time efficient method with smart sampling. For these two reasons we chose to use the Lp-Adaptation method which is described below.

Lp-Adaptation is a novel statistical method [15], that unites approximate design centering and volume estimation. Its process is inspired by the evolution of robustness in biological systems and by randomized schemes for convex volume computation. It is able to address both problems in the general non-convex case and at low computational cost. Lp-Adaptation has been tested on known benchmarks, where it outperformed the previous state of the art in all cases.

The idea of the algorithm is to iteratively sample the parameter space using uniform density over Lp-balls. The Lp-balls dynamically adapt their position, orientation and aspect ratio based on the sampling history. This process is based on the concept of Gaussian Adaptation, in which the mean and the convariance matrix are adapted continuously. Lp-Adaptation requires two inputs: a feasible start point and an oracle. An oracle is a function that takes the candidate points as input and returns if they are in the feasible region of the parameter space or not.

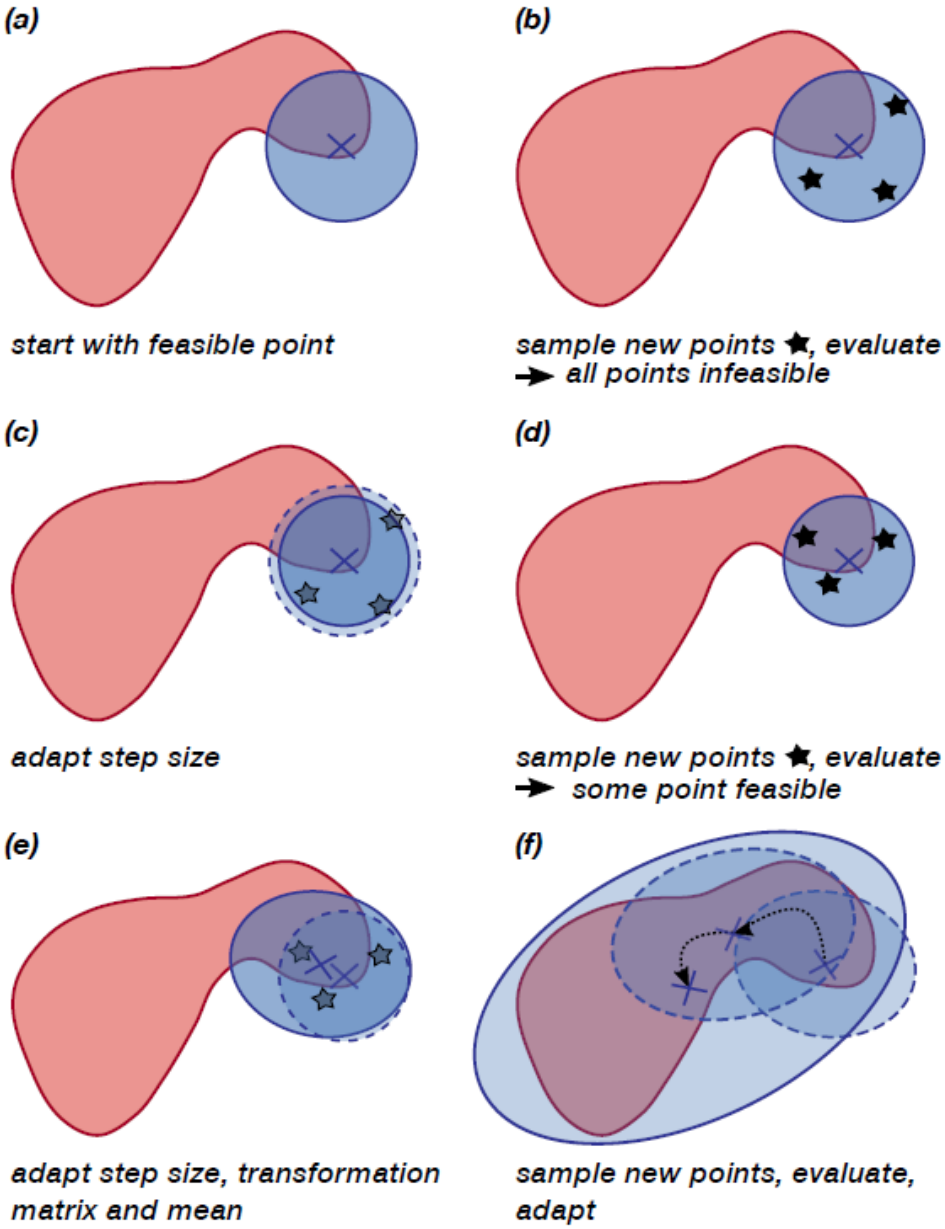


FIGURE 3.1: Schematic illustration of the algorithmic procedure of adapting the proposal distribution (blue ellipse) to the feasible region (red). The cross represents the mean of the proposal distribution. (a) The algorithm requires a feasible point so start with. This feasible point is set as the mean (center) of the initial proposal distribution. The initial proposal ball is isotropic with a radius of 1. (b,d) New points are drawn from the proposal and evaluated against the specifications by querying the membership oracle. (c) If all points are infeasible, the ball radius is reduced in order to increase the probability of sampling a feasible point next. The shape of the proposal remains unchanged. (e) If at least one point is feasible, the location, shape, and radius of the proposal are adapted by moving the mean in the direction of the center of all feasible points, increasing the radius, and adapting the affine transformation to include information about the distribution of feasible points. (f) At the end of the process, the proposal will have the largest possible volume for the given target hitting probability. Now, the mean can be used as a design center, and the volume can be approximated from the determinant of the proposal. (figure reproduced from Asmus et al. 2017 [15] under license: `https://creativecommons.org/licenses/by/4.0/`)

The steps followed by the algorithm are shown below and the detailed algorithmic procedure on Figure 3.1

1. **Initialization** → provide start point and oracle
2. **Sampling** → uniform density over Lp-balls
3. **Evaluation** → based on the oracle
4. **Adaptation** → Gaussian Adaptation

The Lp-Adaptation algorithm comes with some optional parameters such as the p-norm, the starting search radius, the number of points sampled per iteration, the number of evaluations and the target hitting probability P.

Probability P can either have a predefined value or be adapted during the design centering estimation. This is an important feature that allows Lp-Adaptation to approach both the design centering and the volume estimation problem. Low hitting probabilities lead to low sampling efficiency while high hitting probabilities lead to slower adaptation to the feasible region.For a Gaussian proposal and a convex feasible region, the optimal hitting probability is 1/e, but for non-convex cases there is no such information. Therefore, it is suggested to adapt the hitting probability starting from 1/e as an initial value. We can see the effects of different hitting probabilities in Figure 3.2.

Concerning the use of Lp-Adaptation in the bioimage analysis process of this project we had to consider the following matters:

- Lp-Adaptation works in a continuous and unbounded parameter space whereas the parameters of bioimage analysis algorithms can be both discrete and bounded. Discrete parameters are in general easier to tune and we did not use them in the context of this project, but we used bounded ones. To adapt Lp-Adaptation to this problem we made a slight change in the sampling step: every time an out-of-bounds point is sampled, the parameters that exceed the bounds get the value of the closest bound. For example, if the bound is between 0 and 1 and the point (0.5,-2,3) is sampled, it becomes (0.5,0,1).
- Choosing the number of evaluations of the method was a compromise between the amount of exploration of the parameter space and time. As mentioned already, bioimage analysis processes can get costly on time so we had to find the number of evaluations that would be enough for exploring the parameter space without reaching non-practical execution time. We will see more about this trade-off in the 'Time Benchmarks' section 5.7.
- The choice of hitting probability P was also an important matter to consider. Since the convexity or not of the parameters space was not known, we chose to use adapting hitting probability. Using a descending hitting probability is better for volume estimation, while using an ascending one is better for design centering. So we used ascending P probability and tried different values and intervals (further analyzed in section 5).

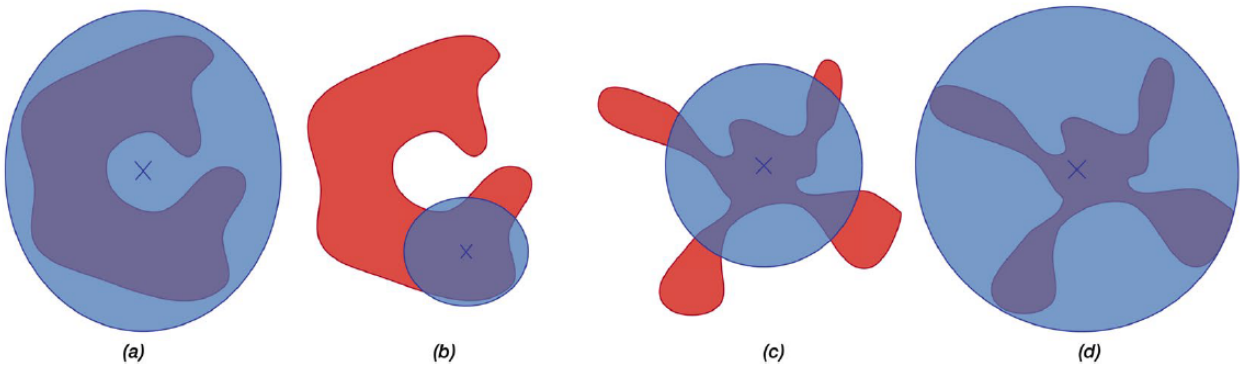


FIGURE 3.2: Illustration of the effect of different hitting probabilities on a feasible region. (a) Low hitting probabilities may lead to infeasible design centers. (b) Increasing the hitting probability (e.g. shrinking the proposal) leads to a feasible design center. (c) For volume estimation, high hitting probabilities may underestimate the volume. (d) Decreasing the hitting probability leads to better volume approximation. (figure reproduced from Asmus et al. 2017 [15] under license: `https://creativecommons.org/licenses/by/4.0/`)

3.2 Fiji Bioimage analysis platform

As mentioned in the previous section, the Lp-Adaptation algorithm needs an oracle as input to function. In our first idea the oracle would be the user, i.e. for each sampled point by the design centering algorithm the user would answer to the question if the segmentation is good or not. But this approach would require the presence of the user for a long time with interrupts in the middle, which did not fit well to the typical workflow of a biologist. A second idea was to make an offline oracle that would be based on information about the image provided by the user. In this way, it would take only a small amount of time and effort by the user to help in building the oracle. In order to do that we designed a user interface which we implemented in the Fiji analysis platform described in this section.

Fiji [28] is a distribution of the popular open-source software ImageJ (known previously as NIH Image) focused on bioimage analysis. It contains a wide collection of plugins that can be used for various bioimage analysis tasks. Nearly all aspects of ImageJ are pluggable which means that plugins can be provided ad hoc to perform specified functions. When a new plugin is developed, it can be shared with the end users through an integrated update system. [28] ImageJ has been in use for over 25 years and still persists to push and drive the image analysis field. [29] One of the main reasons for ImageJ's success is that scientists can use the infrastructure and dissemination it offers while focusing on developing just the application specific algorithm.

ImageJ2, the last version of ImgeJ on which Fiji is based, is built on the SciJava Common plugin framework. A plugin is a Java class that can be automatically discovered and indexed when the application is launched. [30] ImageJ API is very helpful in the plugin creation process as it contains most classes needed for developing an image analysis algorithm and programming the user dialogue. At the same time, ImageJ's main user interface

(Figure A.2) offers a wide range of selection tools (and more) that can be used for any instance of an ImageJ image window without further programming. In our project, this last feature was essential in getting different kinds of input about the image from the user.

3.3 Covariance Matrix Adaptation Evolutionary Strategy (CMAES)

In section 3.1 we saw that the Lp-Adaptation algorithm requires a feasible starting point to begin the **Sampling** and a membership oracle for the **Evaluation** step. Since the oracle would be based on the input from the user, we had to find a way to provide the starting point. The first distinct thought was to directly ask the user for a feasible point. This meant that the user would have to do some manual tuning until reaching a satisfying segmentation result and then use the benefits of the design centering in matters of robustness. This approach could theoretically succeed in giving us a feasible starting point, but since there was still some manual tuning needed, it would only partially solve the problem. The next thought was to use again the oracle, which was basically intended for the evaluation, as a score function. In that way we could approach the starting point estimation as a continuous optimization problem of the score funtion. To solve that problem we decided to use CMA-ES which is described below.

The Covariance Matrix Adaptation [31] [32] is a non-linear non-convex black-box evolutionary optimization algorithm. CMA-ES is considered state-of-the-art in evolutionary computation and is used widely among research and industry applications around the world. The main steps of the algorithm are: 1) sampling of new points, 2) ordering of the sampled points according to their fitness, 3) updating of the internal state variables based on the samples ordering. One of its advantage comparing to other evolution strategies is the use of correlated mutations instead of axis-parallel ones.

We chose evolutionary optimization because of its better performance, compared to other optimization methods, in tuning the parameters of bioimage segmentation applications (as mentioned in section 2.2). Furthermore, the choice of CMA-ES was due to the fact that it is a non-convex algorithm that does not require any parameter tuning (with the exception of the the population size λ), so it fitted very well to the required conditions of our problem.

3.4 Squassh Segmentation Algorithm

For the implementation and testing of Apeba we decided to train the parameters of the Squassh segmentation algorithm.

Squassh [33] (segmentation and quantification of subcellular shapes) is an open-source Fiji and ImageJ plugin for detecting, delineating and quantifying subcellular structures in fluorescence microscopy images. Squassh can be applied to a wide spectrum of cases by changing some user-defined parameters. The algorithm does not make assumptions about the shapes of the objects so the results are not biased. One important advantage of Squassh is that it is not limited to spot-like or spherical objects, but it can also segment more complex shapes.

The segmentation procedure of Squassh has the following steps:

1. Background Subtraction: the rolling-ball algorithm [34] is used to correct uneven background intensity.
2. Object detection: a model-based algorithm is used to find regions in the image that contain objects of interest.
3. Thresholding: the threshold applied is the user-defined minimum object intensity of objects. All objects with peak intensities lower than this threshold are discarded, and the others are retained.
4. Decomposition: the image is decomposed into smaller parts in order to allow different background and foreground for different objects, ensuring that the different parts do not influence each other by computing the Voronoi diagram of the binary mask.
5. Local Background and object intensities: local analysis for each image region and intensity estimation.
6. Individual Object Segmentation: seperate segmentation for each image region with subpixel analysis.
7. The final segmentation: optimization of the previous step by minimizing the segmentation error.

We decided to use Squassh because of its flexibility in different kinds of data and the amount of control that the user-defined parameters have over the segmentation result. The parameters that can be tuned are: 1) the minimum object intensity, 2) the lambda regularization. As mentioned above, the minimum object intensity is used in steps 3 and 6. The values of this parameter are normalized between 0 for the smallest value occurring in the image and 1 for the largest value. By using low values we increase the sensitivity, while with high values we can achieve the separation of objects. Lambda regularization is used in steps 2 and 6, and it can significantly affect the segmentation results. By using high values we can avoid the segmentation of noise. The typical values are between 0.05 and 0.25. While lambda regularization has only a lower bound at 0, we decided to use its values from 0 to 1 to have cohesion with the minimum intensity range but also because values above 1 are not commonly used in practice.

4 Apeba Segmentation Plugin

The Apeba segmentation plugin is an implementation of the concept of automatic and robust parameter tuning introduced in chapter 3. Apeba is implemented for the free open-source bioimage processing framework Fiji. In this chapter we are going to analyze the steps followed by the plugin along with its implementation and code structure.

4.1 Overview

The Apeba plugin was developed in order to perform all the automatic tuning and segmentation process, from the initial image import to the final segmentation result.

Big part of the plugin was the implementation of the oracle. Apart from the UI and the way that we would get the input from the user, we had to consider how the evaluation of the points would be done, i.e. how we would utilize the given information for the image to reach the decision if a sampled segmentation is good or not. In order to do that, we constructed each oracle as a scoring function and used thresholds to get the binary answer 'yes' or 'no'.

In matters of human-computer interaction, we wanted to design the Apeba segmentation plugin in a way that it would be easy to understand and use, guiding smoothly the user through the parameter tuning and segmentation process. The ultimate goal was to design a user interface that could be operated by people from all levels of experience and backgrounds, even without any prior knowledge of bioimage analysis or Fiji applications. Furthermore, we wanted to automate as much the process by giving, wherever it was possible, recommendations of the input needed from the user (more in 4.2.1 and 4.2.4).

The programming languages used were Java and MATLAB. In general, programming an application that combines these two languages can be time consuming while the data transferring through function calling adds extra time to the overall process. But since Fiji plugins can only be programmed in Java and the Lp-Adaptation algorithm is only developed in MATLAB, using both of them was necessary.

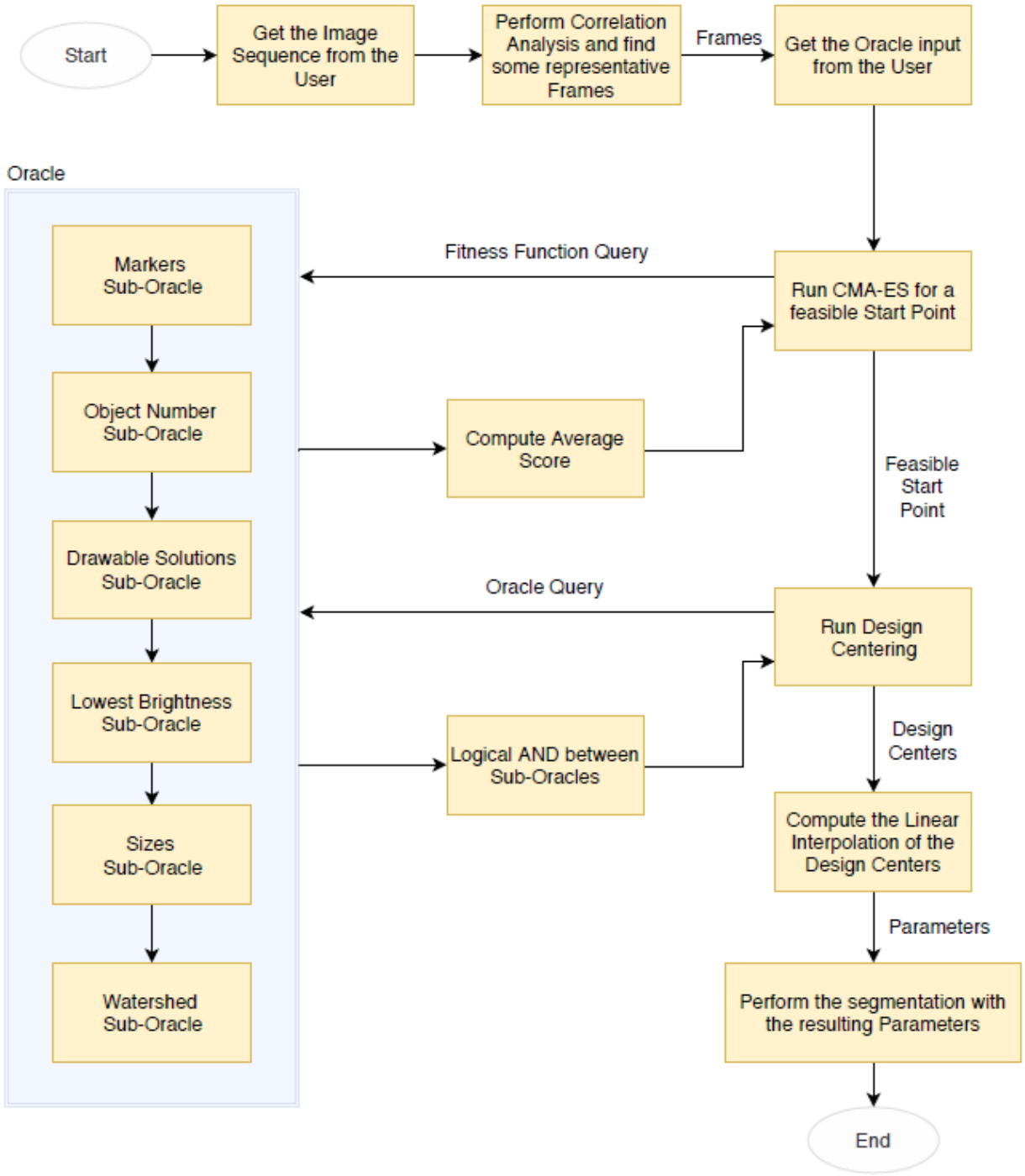


FIGURE 4.1: Flowdiagram of the procedure followed by the Apeba plugin.

Below we can see the main steps of the plugin:

1. **Image Input**: the user chooses the image sequence for the segmentation, then the sequence is imported and opened as a default Fiji image window which is integrated in the UI of the plugin. (Figure A.1)
2. **Key-frame selection**: here the user can choose the key-frames of the image sequence that will be used for the design centering estimation. The plugin also suggests these

keyframes based on a correlation analysis of the frames in the sequence (further explained in section 4.2.1).

3. **Oracle Input from the user**: in this step the user is giving different information about the image, with the help of the UI and the Fiji selection toolbar. The type(s) of input given can be chosen by the user. These information are then used to build an offline oracle (further explained in sections 4.2.2-4.2.8).
4. **Start point estimation**: the oracle is used as the fitness function for the CMA evolution strategy in order to find a feasible point from the parameter space that can be used as input for the Lp-Adaptation algorithm (further analyzed in section 4.2.9).
5. **Design center(s) estimation**: the Lp-Adaptation algorithm is used to find the design center of each key-frame (further analyzed in section 4.2.10).
6. **Final segmentation**: the Squassh segmentation algorithm is executed with parameters that are based on the design centering results. If there was only one key-frame, the estimated design center is used for every frame of the image sequence. If there were multiple key-frames, the parameters follow the linear interpolation of the design centers from each key-frame.

In Figure 4.1 we can see a more detailed flowdiagram of these steps.

4.2 Step by step process

4.2.1 Key-frame selection

As mentioned before, the plugin offers the possibility for single or multi-frame analysis. In the case of multi-frame analysis, the design center is estimated for each key-frame and then the segmentation parameters follow the linear interpolation of the design centers. Hence, the number of key-frames as well as their index can play a major role in the final segmentation result. By letting the user choose the key-frames, we can rely on the human vision system which is superb when it comes to qualitative tasks [1]. Nevertheless, as the sequence size and the changes between the images increase, this task can become very complex and lose its qualitative advantage. For that reason we wanted to give also some key-frame recommendations, depending on the changes throughout the image sequence. Analyzing more key-frames means that we can get a better view of the different states in the image sequence, but this can be costly in time. The user interface of the key-frame selection can be seen in Figure A.3.

The choice of the recommended key-frames is based on an autocorrelation analysis of the given sequence which is done by calculating the Spearman's rank correlation coefficient between the images. The steps of the key-frame choosing process can be seen in Algorithm 1. The correlation is calculated between a *current* frame and its latter ones. Every time the correlation reaches a stable value and starts to fluctuate, the *current* frame is updated. The first key-frame is always the first image of the sequence and the rest of them are selected every time the correlation falls at a certain amount from the correlation of the previous key-frame. The value of this difference that has to be reached in order to select a new key-frame, can be controlled by the user with the "More" and "Less" buttons. In the case of the "More"

button, the value of the difference is decreased resulting in more key-frames, while in the case of "Less" the difference is increased resulting in less key-frames.

In Figure 4.2 we can see the results of running the algorithm in an image sequence of 287 frames. The highlighted bars represent the chosen key-frames. The first key-frame is used as the *current* until the correlation starts to fluctuate and so it is updated at the frame with index 208. The actual flunctuation starts around frame 190 but there is a delay due to the size of the check window (sw=20 in this case). We can also notice that every time the correlation falls by 0.1 from the previous key-frame, a new key-frame is chosen.

Algorithm 1: Using Spearman's correlation for key-frame choosing

Input: Image Sequence L, Size of check window sw, difference df
Output: list of key-frames

```
1  K ← none;
2  current ← first frame of L;
3  lcor ← 1 ;
4  add current to K;
5  for i: number of frames in L do
6      frame ← read frame with index i from L;
7      cor ← spearman(frame, current);
8      if lcor − cor > df then
9          add frame to K;
10         lcor ← cor;
11     if i mod sw = 0 then
12         avgcur: find average correlation between current and frames with indices from
             (i − sw) to i;
13         if avgcur > avgprev then
14             add frame to K;
15             lcor ← cor;
16             current ← frame;
17         avgprev ← avgcur;
```

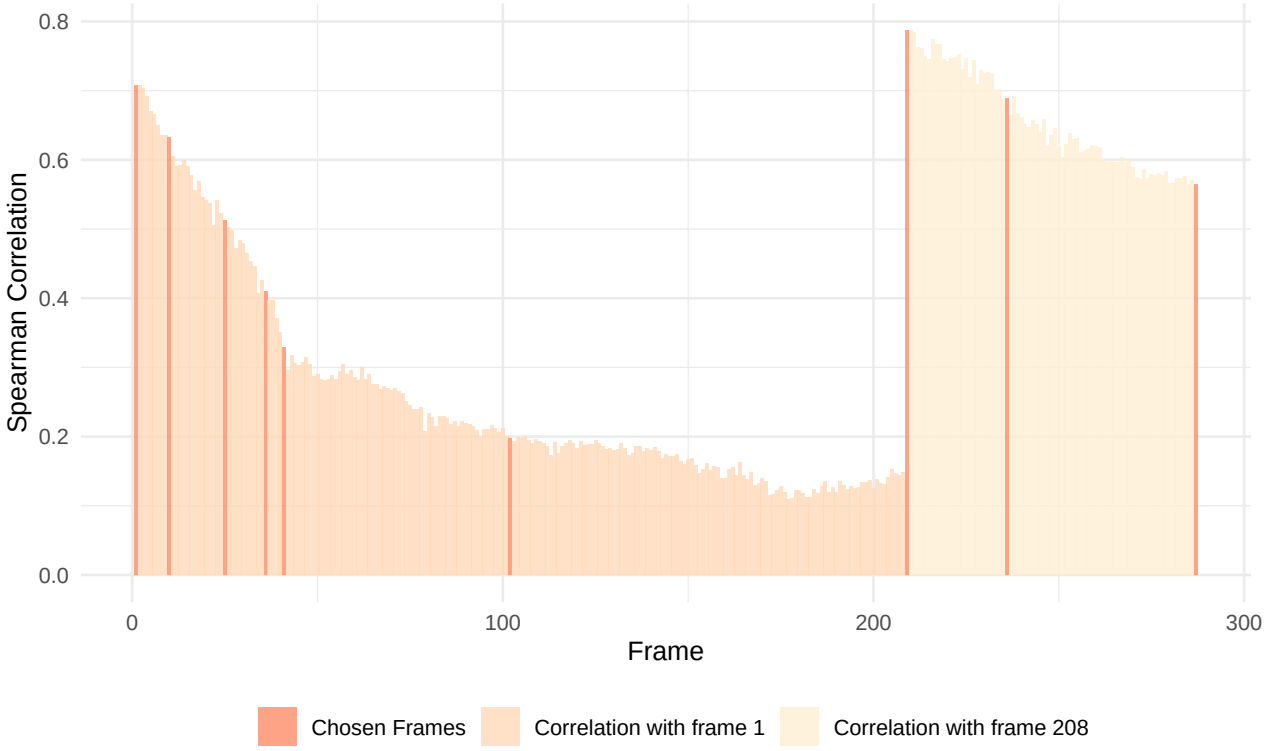


FIGURE 4.2: A key-frame automatic selection example using the Algortihm 1.

4.2.2 Sub-oracles selection

After the key-frame selection, the user has to choose the different ways to provide input for the images. In the UI (Figure A.4), this is done by clicking the names of the desired sub-oracles in a checklist. It is possible to choose from only one of them to all. The choice depends on the information that the user is able to provide about the image but also on the segmentation task. For example, the object number sub-oracle cannot be chosen for images with very big cell count since it would take a lot of time and effort to count all the cells. Later we will see more examples about the cases to use each sub-oracle.

4.2.3 Markers sub-oracle

In this oracle the user can provide input for the image by putting a marker to each object. This is done with the help of the Fiji 'Multi-Point' tool which is preselected. We can see the UI and an example of this sub-oracle in Figure A.5.

The score of this sub-oracle is the count of the correctly segmented markers divided by the total count of given markers. A marker is considered to be correctly segmented if 1) it is included in the segmentation mask of an object and 2) there was no other marker previously found for this object. For example, if the user has marked two different objects that are segmented as one, only one of the two markers will be considered correctly segmented. The steps for doing the above can be seen in Algorithm 2.

In general, it is recommended for the user to try to put the markers in the center of each object to ensure that it will not be mistaken for another segmented area that is close. Also,

it is not necessary to have a marker for every object of the image, which allows to use this method even for images with big cell count.

Algorithm 2: Calculate Markers sub-oracle score

Input: Markers coordinates X, Segmentation mask M
Output: Percentage of successfully segmented markers

```
1 P ← none;
2 c ← 0;
3 for number of markers in X do
4     read current marker;
5     if marker belongs to a segmented object in M AND object ∉ P then
6         c ← c + 1;
7         add object to P;
8     score ← c / number of markers in X;
```

4.2.4 Object Number sub-oracle

In this sub-oracle the user gives the object count that the desired segmentation should have. For this method we also provide a recommendation for the count which is found by the number of markers given in the previous sub-oracle (if it was chosen). This method can play an important role in defining the "good" segmentation, especially if it is combined with the Markers one because you can make sure that all the objects will be segmented and they will be located in the right positions. Therefore, it is recommended to use it in most cases unless it is not possible to provide such input (e.g. in case of a very large object count that is difficult to be measured by a human). We can see how the UI looks for this method in Figure A.6. The text field is already filled by a recommendation count when the window opens that the user can keep or change.

For the evaluation in this case we just calculate the count of segmented of objects over the given count by the user:

$$score = \frac{\text{count of segmented objects}}{\text{ground truth count}} \tag{4.1}$$

4.2.5 Drawable Solutions sub-oracle

In this sub-oracle the user has to "draw" the segmentation. More specifically, the user must outline the objects that should be segmented. It is not necessary to draw the whole segmentation, and in most cases even providing input for a small part of it can be proven really helpful for the oracle. For example, in the case that we have different kinds of cells, it can be sufficient to only provide one outline from each cell type. This kind of input can be valuable in avoiding overfitting or underfitting in the segmentation process and in tuning parameters like the threshold. That is because we have an exact ground truth of the outline of an object and so we can reject segmentations that have different boundaries. In Figure A.7 we can see how the UI looks and an example of a partially drawn solution. The outline of the objects is done with the help of the "Freehand selections" tool.

The evaluation in this case is done by calculating the Jaccard similarity between the sampled segmentation mask and the drawn one by the user. This is done separately for each object outlined. The step-by-step process can be seen in algorithm 3. The final score is calculated as the average similarity for all the objects over their count. If we want to focus more on the size of the segmentation areas, an alternative to the latter would be to use the percentage of the area covered by each object as their weight in the scoring formula.

This oracle can provide valuable information about the exact boundaries of the objects of the segmentation, so it is important for the user to be as accurate as possible. This also means that the use of this sub-oracle should be avoided if the segmentation boundaries are not important, e.g. in the case of an object count task, as it can influence the overall fitting of the parameters.

Algorithm 3: Calculate Drawable Solutions sub-oracle score

Input: Segmentation mask M, list of object drawings D
Output: Average similarity of the objects

1 $score \leftarrow 0$;
2 **for** *number of objects in D* **do**
3 $\quad drawing \leftarrow$ current element of D;
4 $\quad ground \leftarrow$ the object of M that covers the most part of drawing;
5 $\quad score \leftarrow score + similarity(ground, drawing)$;
6 $score \leftarrow \frac{score}{\text{number of objects in D}}$;

4.2.6 Lowest Brightness sub-oracle

In this sub-oracle the user has to provide the area(s) of the segmentation with the lowest brightness, by drawing some rectangles in the image. We can see how the UI looks in Figure A.8. Drawing the rectangles is an easy task to do with the help of the 'Rectangle' selection tool of Fiji.

Bearing in mind that objects with low brightness are the ones that are often ignored by segmentation algorithms, this method helps in segmenting these 'tricky' cases. The evaluation of this sub-oracle is done by calculating the percentage of pixels from each rectangle that is included in the segmentation mask, and then by finding the average score for all the given rectangles. It is important to note that due to the way that this method works, it can also be used to indicate areas that are difficult to segment for other reasons or that is very important to be included in the segmentation.

4.2.7 Sizes sub-oracle

In this sub-oracle the user provides an outline of the smallest and biggest object in the image. We can see how the UI looks in Figure A.9. Again, the tool that is used to outline the object is the "Freehand selections" tool.

The evaluation for this sub-oracle is done by finding the sizes of the smallest and biggest object and then by using them as limits for the segmented objects' size. This means that every object from the sampled segmentation that is out of these limits, is considered as a segmentation miss. Then the sum of these misses is subtracted from the total number of

objects found and the score is calculated as the quotient of this number and the total count (see Algorithm 4). This method can be very helpful in rejecting artifacts that often have very small sizes but also to avoid the segmentation of individual objects as one (e.g. if the objects are very close or touching each other). Contrary to the drawable solutions, the detail of the boundaries in this case does not affect so much the final result, so usually the user can give an approximation of the outlines without compromising on accuracy.

Algorithm 4: Calculate Sizes sub-oracle score

Input: List of sizes of the segmentation objects M, size of big object b, size of small object s, object count c

```
missed ← 0;
for number of objects in M do
    sizecur: get size of current object of M;
    if sizecur > b OR sizecur < s then
        missed ← missed + 1;
score ← (c − missed) / c;
```

4.2.8 Watershed sub-oracle

In this sub-oracle the user indicates with a line the watershed between pairs of touching objects that should be segmented separately. We can see the UI for the Watershed sub-oracle in Figure A.10.

The evaluation for this sub-oracle is done by finding the count of correctly segmented watersheds and divide it by the sum of the watersheds given by the user. The user has to be careful about when choosing this oracle as there are several algorithms that cannot solve watershed problems. Squassh is also one of these algorithms and so the watershed sub-oracle was not used in the testing of the plugin in the context of this project.

4.2.9 CMA-ES for starting point

In section 3.3 we explained the CMA evolution strategy and its use for finding a feasible point. This feasible point is used later as the starting point for the Lp-Adaptation algorithm. Here we are going to talk about the fitness function and the stopping conditions that we used for running CMA-ES.

The fitness function is based on the information about the image given by the user. More specifically, the output of the function is the average score of the sub-oracles that were described above. This score has a minimum value of 0 and a maximum of 1, and the aim of the optimization by CMA-ES is to maximize it.

Concerning the termination of CMA-ES, we had to find a fitness value that would be used as the stopping condition. Since the maximum value that the fitness function takes is different for each case and unknown, using a static fitness value would be problematic. That is because there might be cases where there are no parameters that can reach the fitness value, and so CMA-ES would fail in finding a start point. Of course even this could be valuable information about the capabilities of the algorithm in relation to the segmentation

task. However, bearing in mind that it is hard to say if a score is high or low, as it really depends on the user input, we wanted to focus on finding an acceptable solution irrespective of its fitness. The steps followed to do that can be seen in Algorithm 5. The basic idea is to guide CMA-ES to the solution by using an increasing fitness value. The increase of the fitness value continues as long as CMA-ES finds a solution, and it stops when a pre-defined maximum number of evaluations is reached. If that is the case, then the best point found so far is used as the start point of Lp-Adaptation.

Algorithm 5: CMAES threshold updating

Input: List of thresholds L, Maximum threshold max

```
threshold ← first element of L;
stopflag ← run CMAES(threshold);
while stopflag ="fitness" AND threshold < max do
    threshold ← next element of L;
    stopflag ← run CMAES(threshold);
```

4.2.10 Lp-Adaptation for design centering and segmentation result

As already mentioned, the design center estimation is done by the Lp-Adaptation algorithm. The oracle of Lp-Adaptation is constructed as the combination of the sub-oracles that were chosen by the user. The evaluation is done by using the highest fitness value that was found by CMA-ES as the threshold of the sub-oracles. Then, the final oracle answer arises from the logical AND join of the sub-oracles' thresholded results. This means that even if just one sub-oracle has a lower score than the threshold, the final answer will be "no", i.e. the sampled segmentation is not good.

4.3 Code Structure

In this section we are going to see the structure of the code for the Apeba plugin. As we mentioned, one part of the code was in Java and the other was in MATLAB.

Apart from the pre-existing CMA-ES and Lp-Adaptation functions, the MATLAB code consisted of a main function, an oracle function for Lp-Adaptation and a fitness function for CMAES. The main function gets as input the information for the sub-oracles and can be called by Java for the design centering estimation. It returns the design centering parameters, the volume of the feasible region and the start point used.

The class diagram of the Java code can be seen in Figure 4.3. The structure was generally based on the open-source code of the Weka Segmentation plugin [13]. The "Apeba_Segmentation" is the main class where all the basic operations start by calling methods from other classes. This class along with the "CustomWindow" and "CustomCanvas" are responsible for the UI construction of the plugin. The "ApebaSegmentation" class is a structure for saving the input given by the users and getting them whenever needed. The data for each sub-oracle are saved in lists of ROIs (the Fiji Region Of Interest). The "Squassh_Runner" class is responsible for running the Squassh plugin in different ways, such as for a single design center or multiple ones. The "MatlabOperator" class is used for calling the Lp-Adaptation algorithm from MATLAB and getting the design centering results. To do that, we are using the

MatlabeEngine class that contains methods for starting and connecting to MATLAB and for evaluating MATLAB functions. The "DataSaver" class contains methods for exporting images and the data given by the user to files. By saving these data it is possible to use them for re-running the plugin and skipping the user input part. The "ColocalisationAnalyzer" class is called for finding the key-frame recommendation based on the Spearman correlation. Finally, the "ImageHelper" class can be used for image operations such as seperating an image stack to individual images.

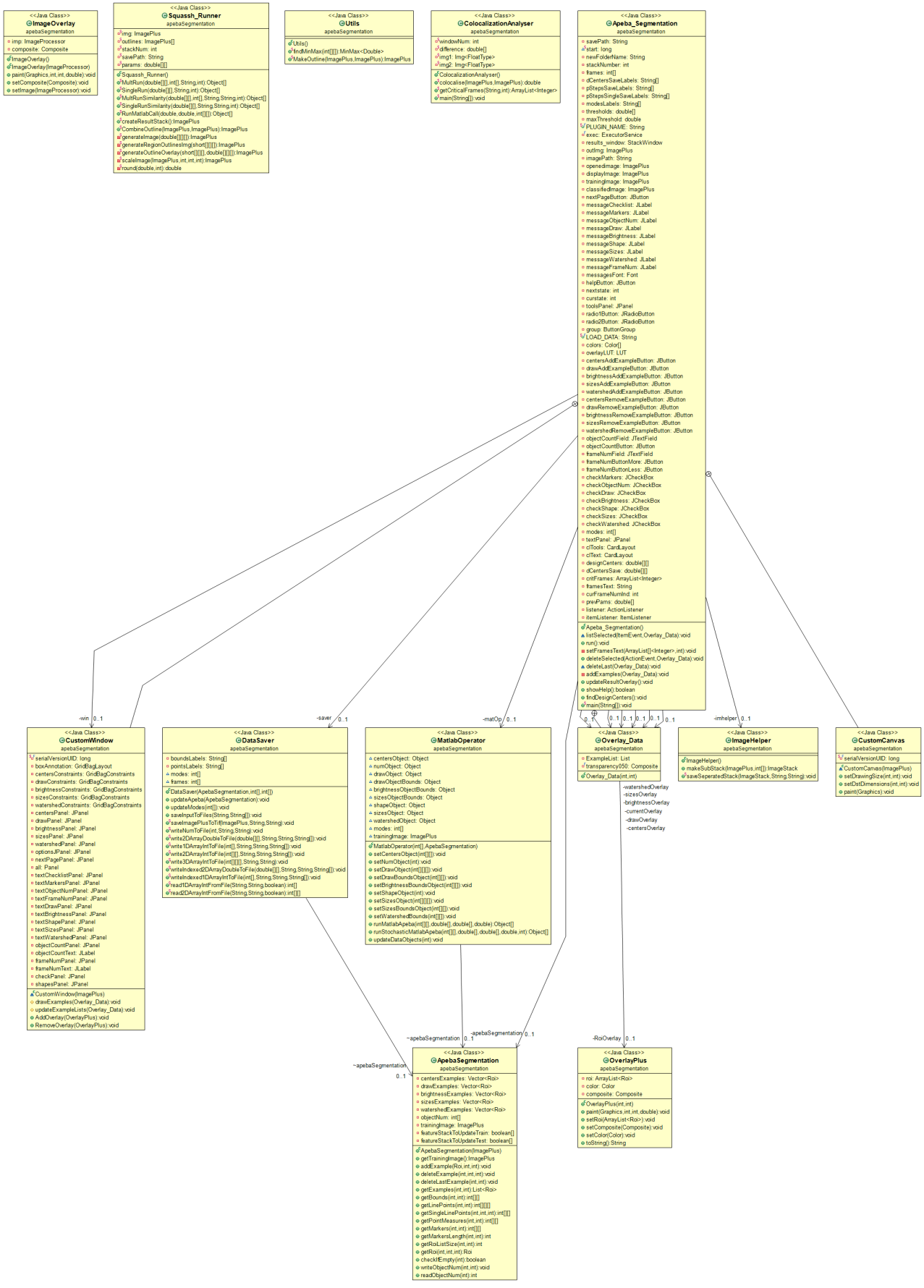


FIGURE 4.3: Class Diagram of the Java code for the Apeba plugin

5 Results

In this chapter we are going to present the results of the different tests that we carried out for the Apeba plugin and the parameter learning process that was earlier introduced. In the first section 5.1 we want to find out how well the design centering problem would succeed in finding a robust solution against the different variations that the biological data can have. In section 5.2 we present some benchmarks of the plugin in matters of quality and time. Finally, in section 5.3 we demonstrate the results of a usability test that was performed and included the use of manual and automating parameter tuning.

5.1 Robustness Against Data Variations

In this section we are going to present the results from tests of the automatic tuning process in image collections with different kinds of variations. More specifically, we used datasets from these four types of image collections: 1) different fields of view, 2) time sequence, 3) different stains, 4) z-stack. The training and the evaluation is based on ground truth data that can be either cell counts or foreground/background segmentation masks. The data for the tests are taken from the "Broad Bioimage Benchmark Collection" [35] and the "Cell Image Library" [36].

We compare three different ways of parameter tuning of the Squassh algorithm: 1) manual tuning, 2) optimization automatic tuning, 3) design centering automatic tuning. The manual tuning is done by one person using the original Squassh plugin for Fiji, the optimization tuning is done by the CMA evolution strategy and the design centering tuning is done by the Lp-Adaptation algorithm using the CMA-ES result as start point.

The parameters used for the CMA-ES and Lp-Adaptation are the same for all the tests. The maximum number of evaluations for each iteration of CMA-ES is 120. The number of evaluations of Lp-Adaptation is 300 while the hitting probability P is increasing following these values: [0.35,0.55,0.75,0.9] with intervals of 1/4. The population size for each iteration is 6.

5.1.1 Variations in Field of View

Object count as ground truth

For our first test we used the "Drosophila Kc167 cells" image set BBBC002v1 [12] from the Broad Bioimage Benchmark collection [37]. This collection consists of five different samples of Drosophila melanogaster Kc167 cells which are stained with a DNA stain Hoechst 33342. Each of the first four samples has a different gene knocked down by RNAi (labeled as 48, 340, Anillin and mad2), while the last sample (labeled as nodsRNA) is of wild-type cells. There are 10 fields of view for each sample, for a total of 50 fields of view. The images

were acquired on a Zeiss Axiovert 200M microscope and their size is 512 x 512 pixels. The ground truth for this image set is the cell count that was determined by two different human counters. It's important to note that the counts of the human observers vary by 16%.

In order to compare the different tuning methods we compute the mean cell count of the 10 images of each sample. Then, we calculate the absolute difference between this mean and the average of the humans' counts for the sample, and divide by the latter to obtain the deviation from the ground truth. After that, we also calculate the percentage of difference from manual and CMA-ES to design centering. These numbers are presented on table 5.1. Concerning the last two columns, when the percentage is negative it means that the deviation decreased and that the design centering tuning had better performance.

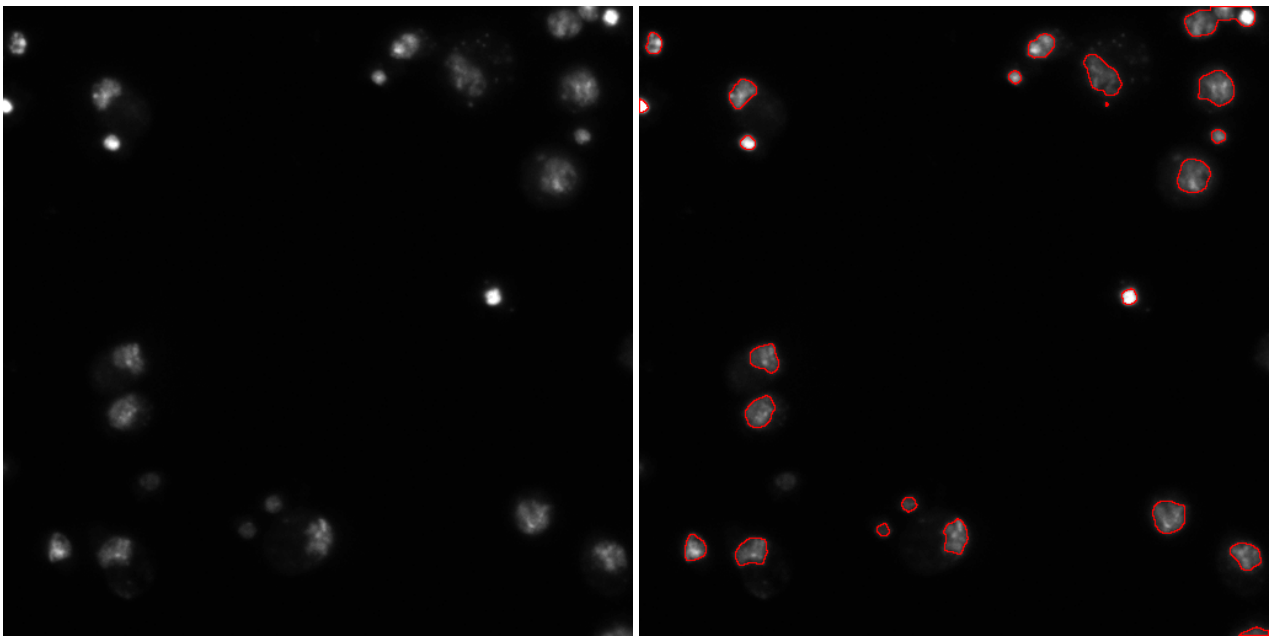
The results for the first sample, labeled as "48", are presented in Figure 5.1. In (A) we can see one frame from this sample and in (B) how the segmentation looks, performed with the design centering parameters. It is possible to observe that this sample contains a small number of cells but their variation in intensity can be quite high. In (C) we can see the plot of the cell count found for each tuning method for all the fields of view of the sample. The gray lines show the count by the two human counters, so the more the results are located between these lines the better. The dots show the key-frames that were used for the training step. So in this case the training was done on the first frame of the image set. In general, we can notice that the lines are for most frames between the counters' values. The manual tuning (green line) seems to have the biggest deviation, which can also be seen from the table 5.1, as its deviation is 13.44%. Also, the design centering method in this case reduces the deviation of the manual one by 38.5%. But when it comes to the CMA-ES, it has a smaller deviation by 17.9%. So the design centering method had better results than the manual one but worse than the optimization.

The rest of the samples are presented in the same way in Figures 5.2, 5.3, 5.4 and 5.5. For the second sample, in which the cells are less but bigger than the first, the design centering method has worse results from both the manual tuning and CMA-ES. But for the rest of the samples it always has better results. One reason for this could be that the later samples have higher number of cells and so the variation might be also higher between the different frames. Furthermore, it is interesting to observe in the 4th sample (Figure 5.4) how the design centering method manages to follow the values by the human counters but the outlines of the segmentation in (B) are way too overfitted to the cells, which means that this segmentation would probably be considered bad if the aim was to have good boundaries of the objects. Overall, we can see that on average design centering reduced the deviation of the manual tuning by 22.8% and of CMAES by 5.46%.

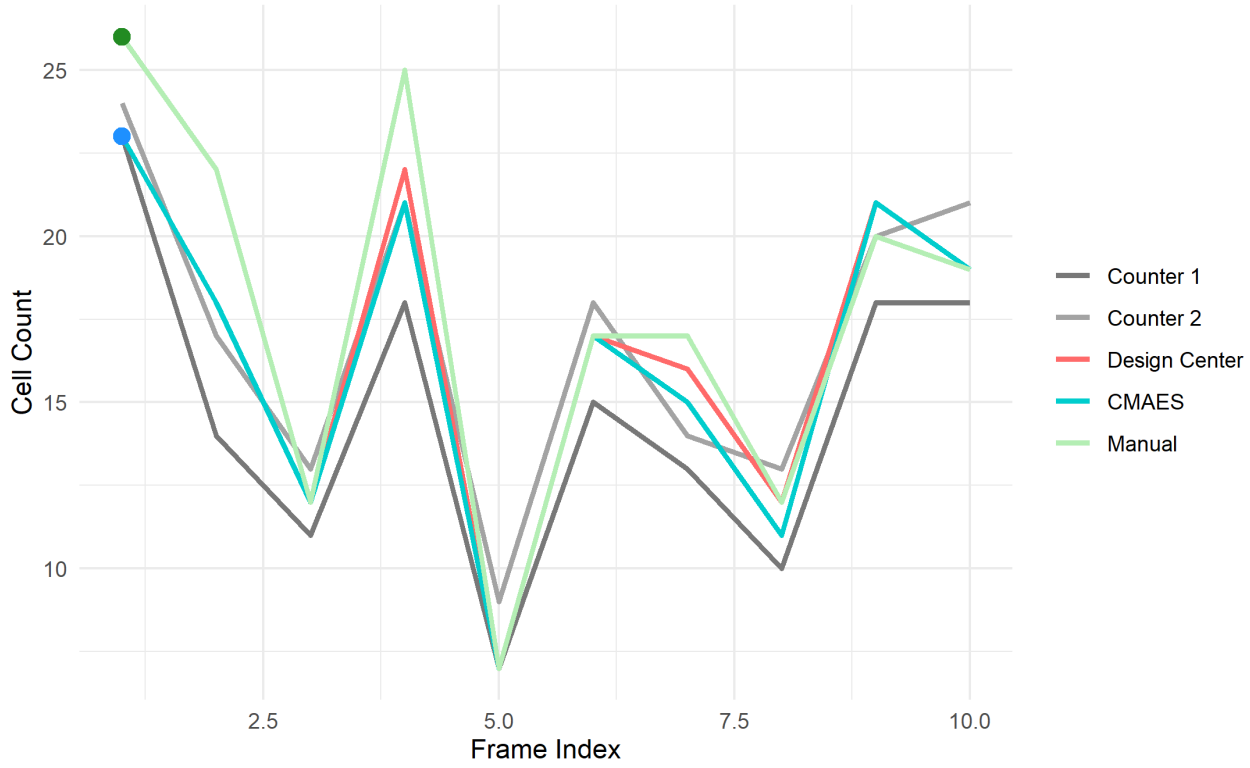
This image set has also been used before by Carpenter et al. [12] with CellProfiler, so we can compare the results. The final result is computed as the mean of the values presented in table 5.1 over all 5 samples. The result from Carpenter et al. is 17% and the results from Squassh are 12.6% for the manual tuning, 10.5% for optimization automatic tuning and 9.6% for the design centering automatic tuning. Apparently, Squassh gets a smaller deviation even by the manual tuning of the parameters, but the addition of the automatic tuning gives an even better result.

Label	Average Deviation from ground truth(%)			Average Design Centering Difference	
	Manual	CMAES	Design Center	from Manual	from CMAES
droskc167s1	13.44	7.00	8.25	-38.5%	17.9%
droskc167s2	15.27	11.72	17.31	13.3%	47.7%
droskc167s3	9.05	14.11	7.40	-8.7%	-41.5%
droskc167s4	17.58	11.20	10.08	-42.6%	-10.0%
droskc167s5	7.86	8.73	5.12	-34.9%	-41.4%
Average	12.64	10.55	9.63	-22.8%	-5.46%

TABLE 5.1: The performance results for the "Drosophilia Kc167 cells" image set.

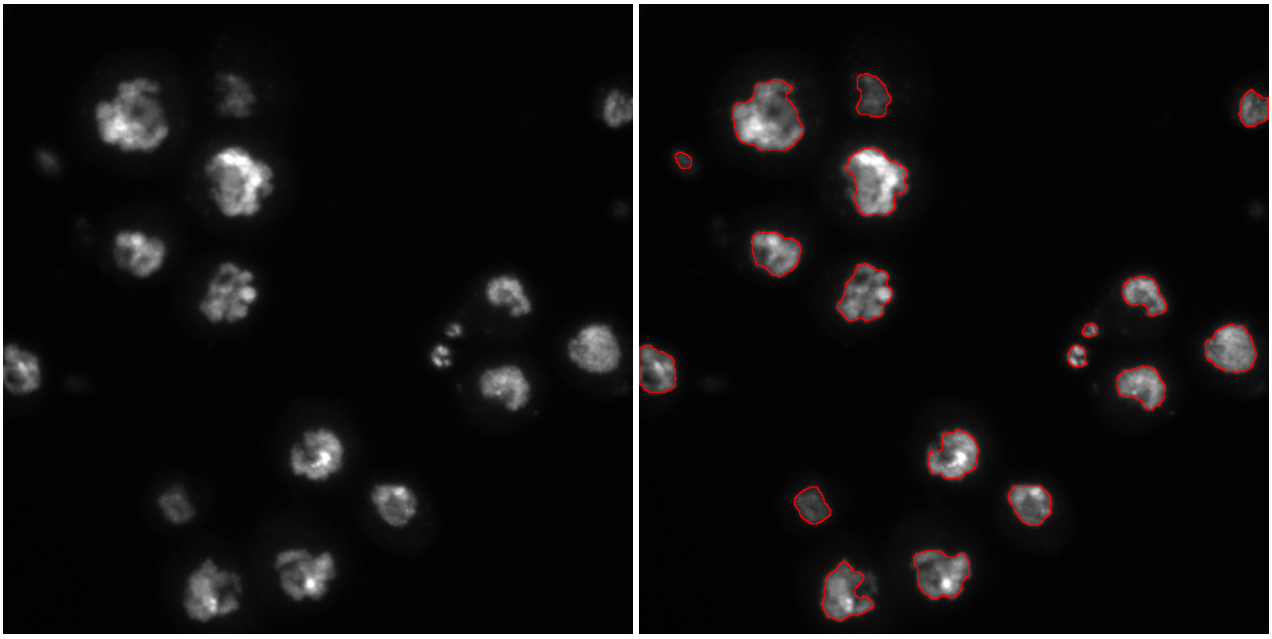


(A) (B)

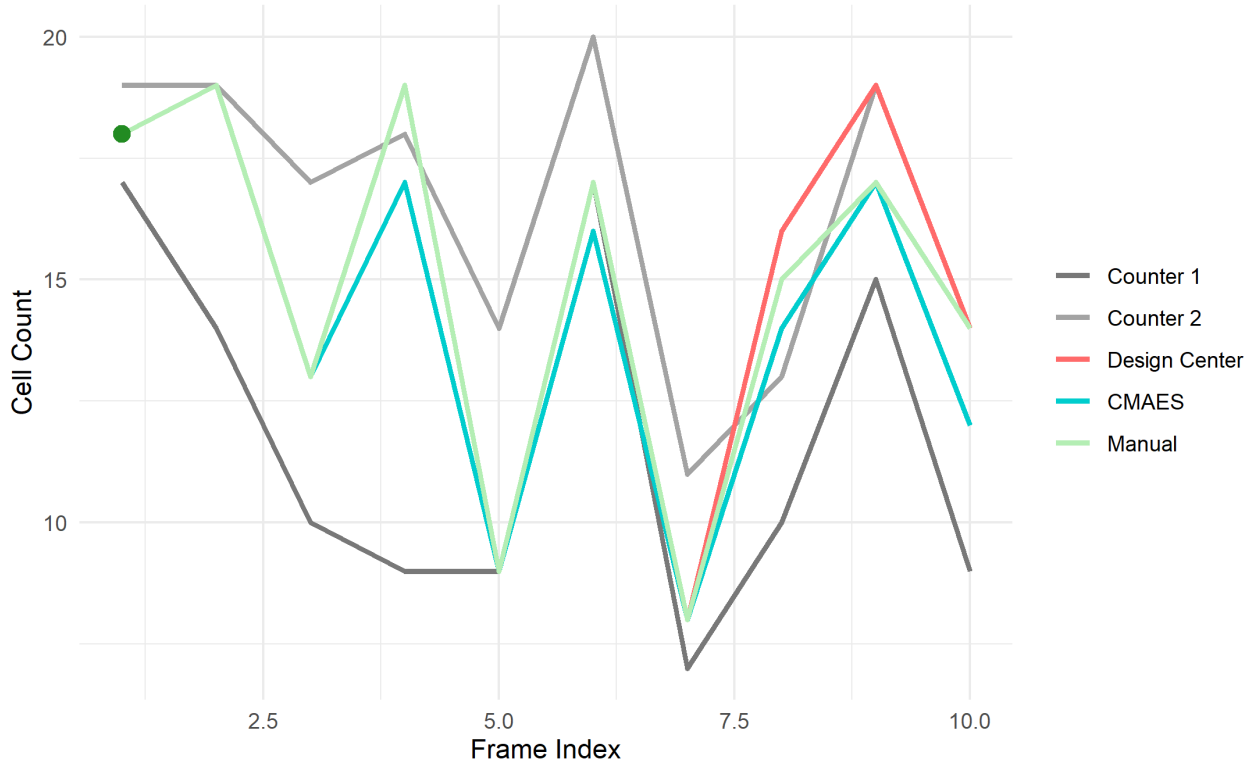


(C)

FIGURE 5.1: The Sample 1 of the "Drosophilia Kc167 cells" image set labeled as '48'. (A) The first frame of the sample. (B) The result of the design centering segmentation of the first frame of the sample. The outlines of the cells can be seen in red. (C) The plot of the cell count results for each frame of the sample.

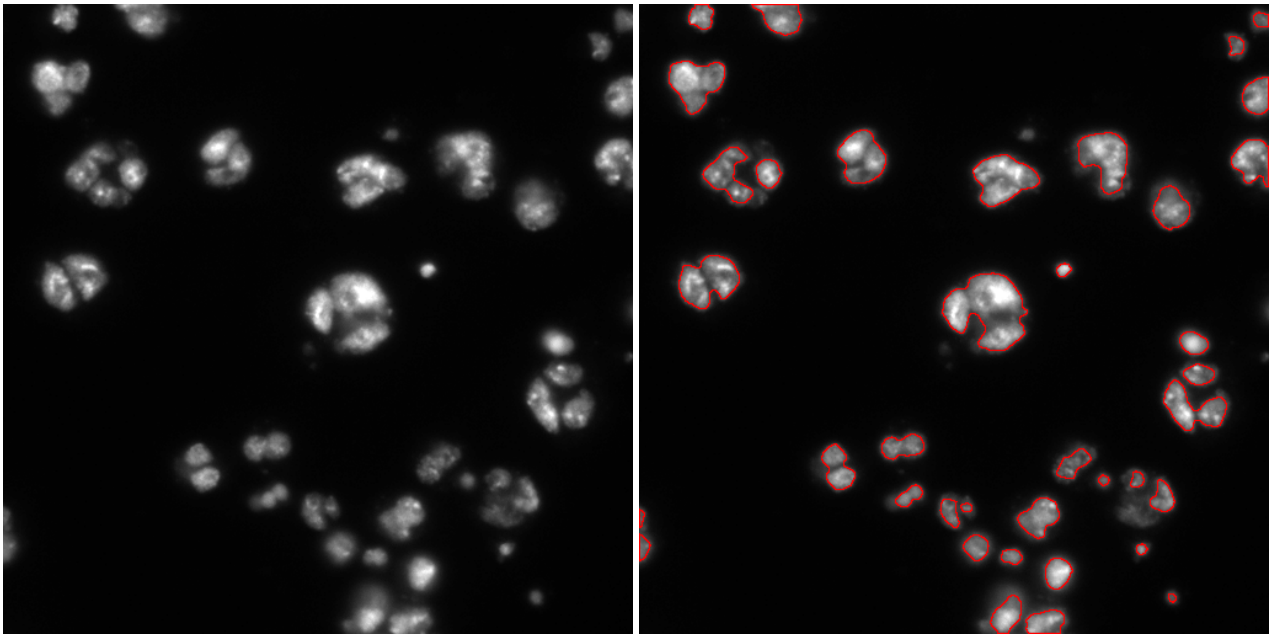


(A) (B)

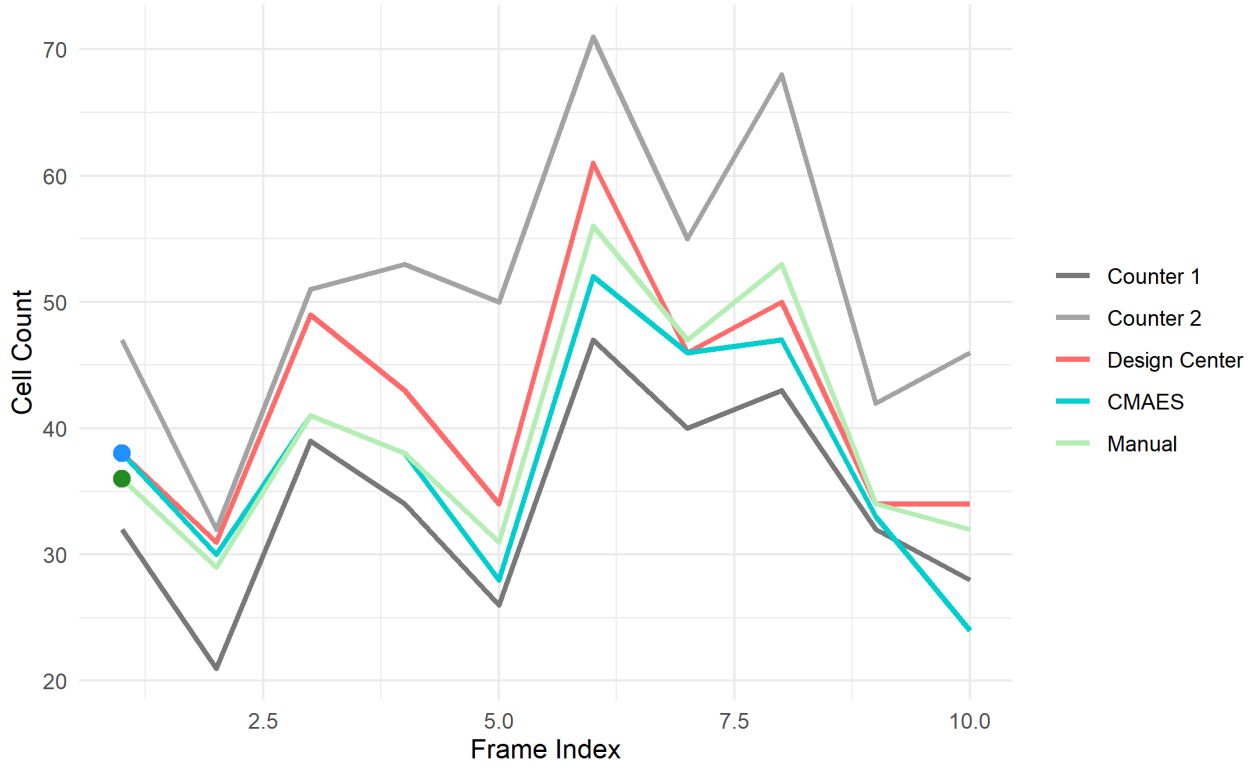


(C)

FIGURE 5.2: The Sample 2 of the "Drosophilia Kc167 cells" image set labeled as '340'. (A) The first frame of the sample. (B) The result of the design centering segmentation of the first frame of the sample. (C) The plot of the cell count results for each frame of the sample.

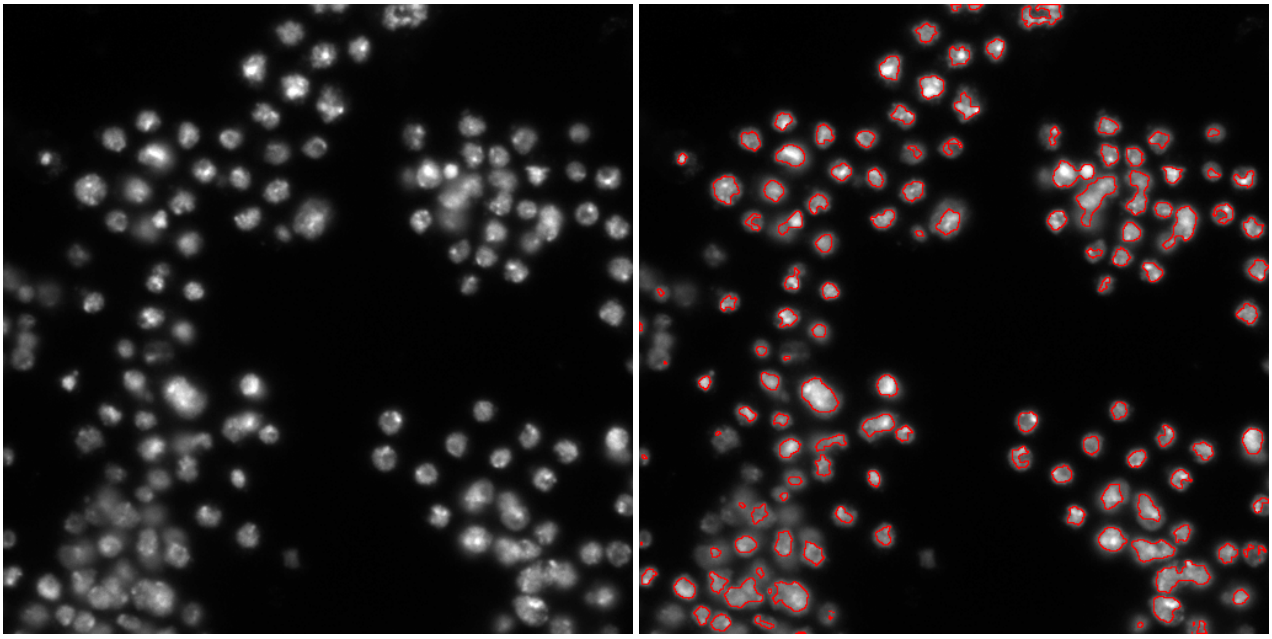


(A) (B)



(C)

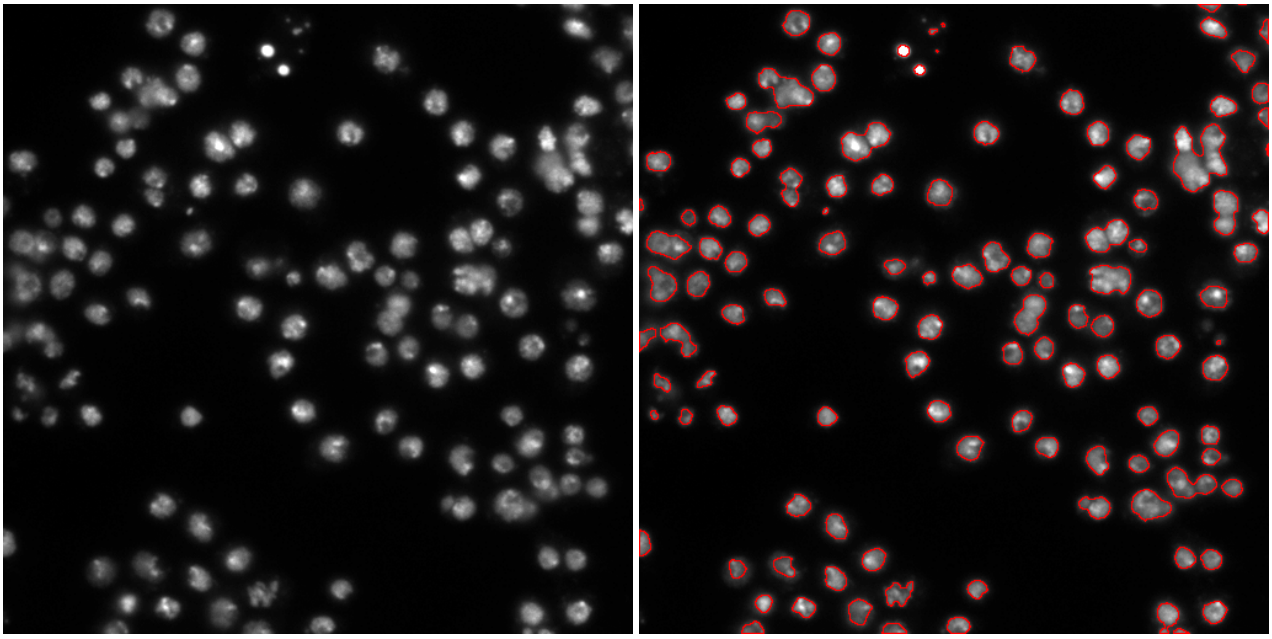
FIGURE 5.3: The Sample 3 of the "Drosophilia Kc167 cells" image set labeled as 'Anallin'. (A) The first frame of the sample. (B) The result of the design centering segmentation of the first frame of the sample. (C) The plot of the cell count results for each frame of the sample.



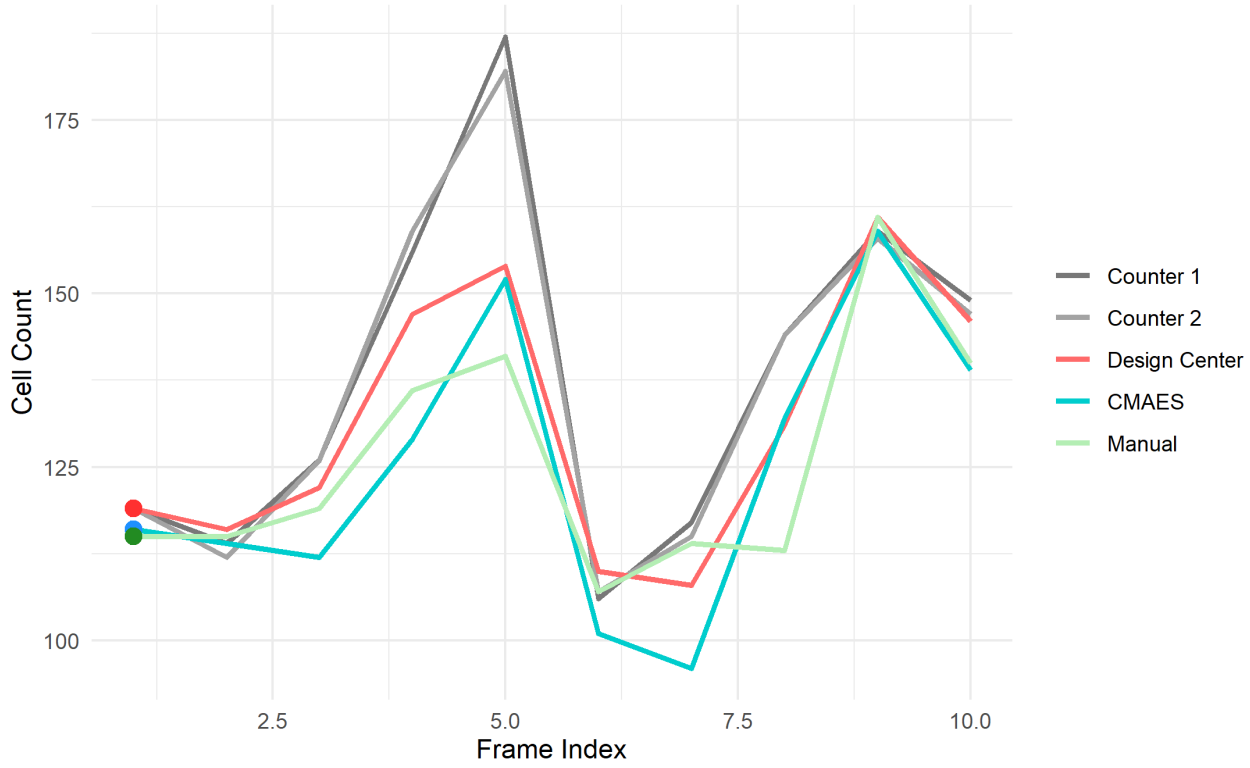
(A) (B)

(C)

FIGURE 5.4: The Sample 4 of the "Drosophilia Kc167 cells" image set labeled as 'mad2'. (A) The first frame of the sample. (B) The result of the design centering segmentation of the first frame of the sample. (C) The plot of the cell count results for each frame of the sample.



(A) (B)



(C)

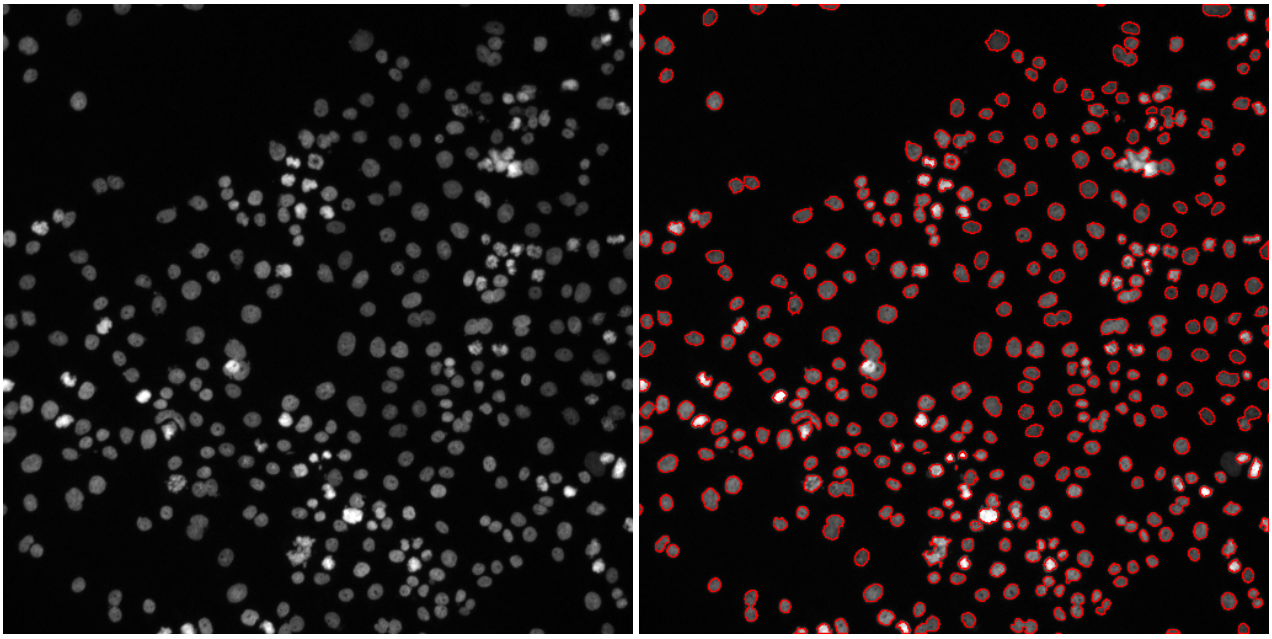
FIGURE 5.5: The Sample 5 of the "Drosophilia Kc167 cells" image set labeled as 'nodsRNA'. (A) The first frame of the sample. (B) The result of the design centering segmentation of the first frame of the sample. (C) The plot of the cell count results for each frame of the sample.

For the second test of the cell count ground truth we used the "Human HT29 colon-cancer cells" image set BBBC001 [12] from the Broad Bioimage Benchmark collection [37]. This collection consists of 6 different fields of view of human HT29 colon cancer cells, a cell line that has been widely used for the study of many normal and neoplastic processes. Usually their analyses follow the common pattern of identifying and counting cells with a phenotype of interest and then normalizing the count by dividing by the total number of cells. So these experiments depend on accurate cell counts. The images were acquired at the Whitehead-MIT Bioimaging Center on a Cellomics ArrayScan and their size is 512 x 512 pixels.

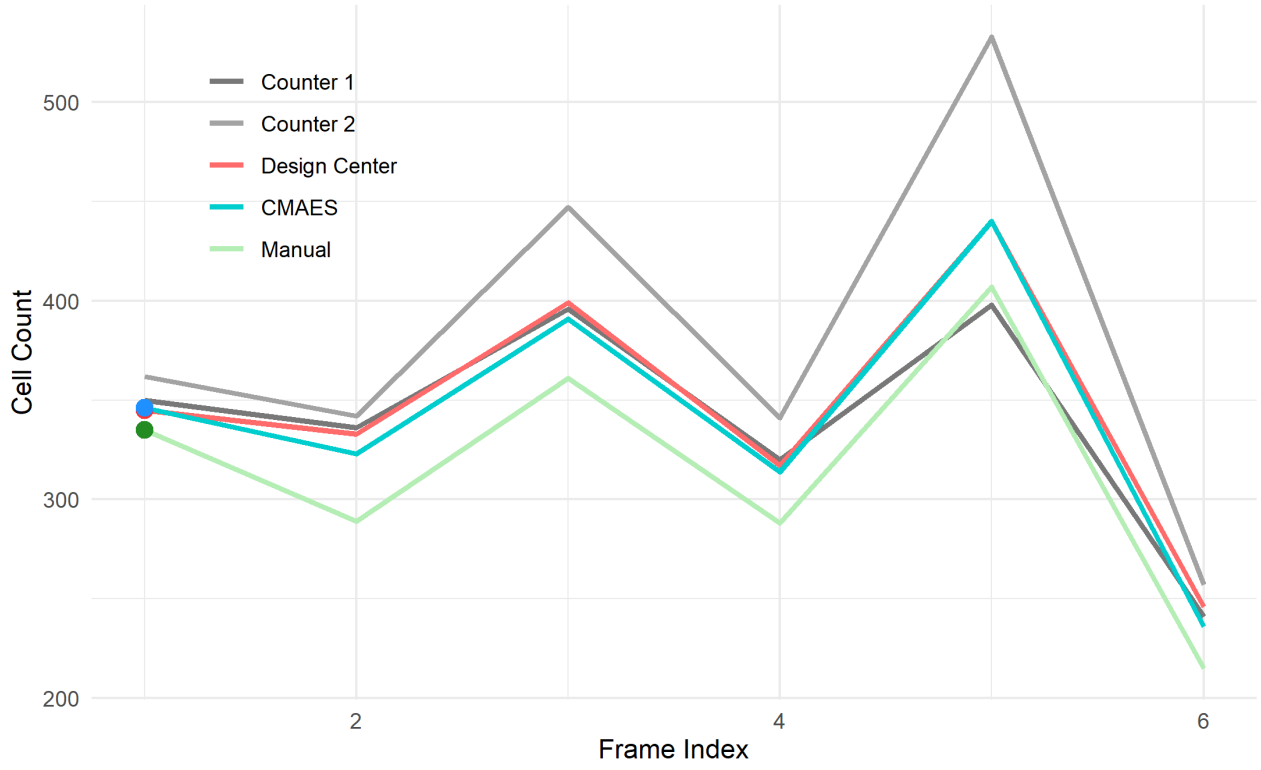
In Figure 5.6 we can see the plot of the cell count results and an image example with and without the segmentation outlines' layer. In this case, the variation between the counters is 11%. The average segmentation results for the frames can be seen in table 5.2. Again the design centering automatic tuning has better results than the manual and CMA-ES, and the deviation decrease in this case is even bigger, 71.7% for the manual tuning and 31.2% for CMAES.

Label	Average Deviation from ground truth(%)			Average Design Centering Difference	
	Manual	CMAES	Design Center	from Manual	from CMAES
humanht29bbbc001	12.34	5.07	3.49	-71.7%	-31.2%

TABLE 5.2: The performance results for the "Human HT29 colon-cancer cells" BBBC001 image set.



(A) (B)



(C)

FIGURE 5.6: The results for the "Human HT29 colon-cancer cells" BBBC001 image set. (A) The first frame of the image set. (B) The result of the design centering segmentation of the first frame. (C) The plot of the cell count results for each frame.

Similarity as ground truth

Some of the image sets in the Broad Bioimage Benchmark collection have the foreground/background segmentation mask as ground truth. The evaluation for this type of ground truth can be done with the help of the Jaccard Similarity (see section 2.1.1).

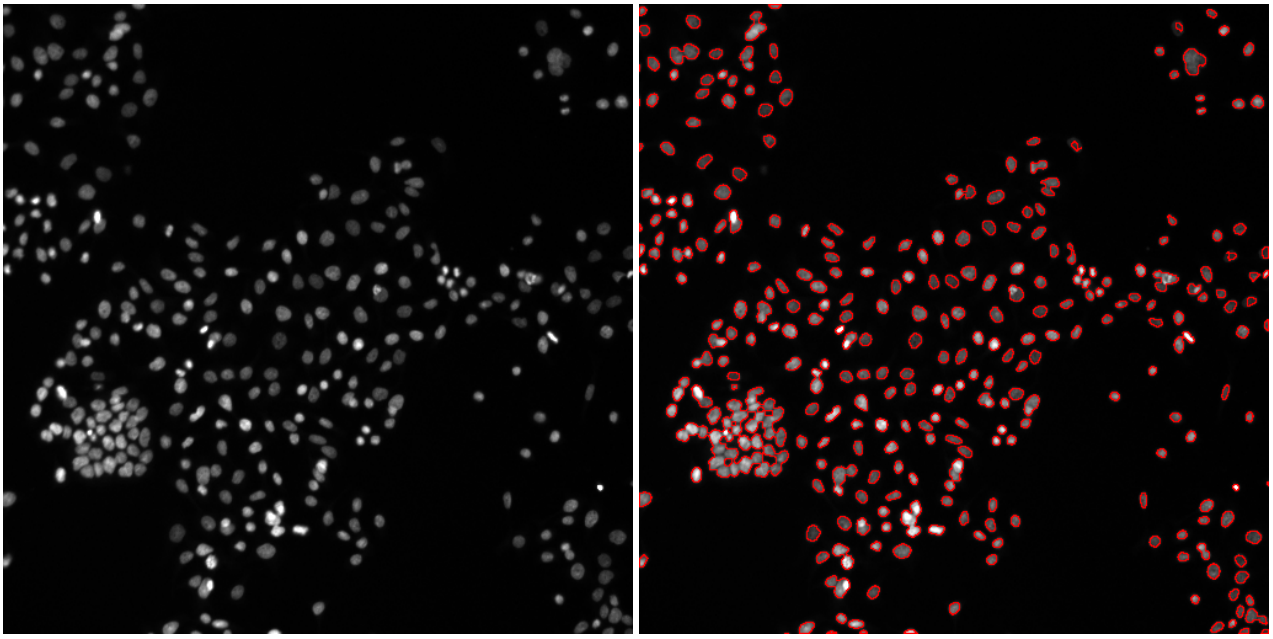
For this test we used the "Human HT29 colon-cancer cells" image set BBBC008. The set consists of 12 images for two different stains, with Hoechst and with phalloidin. Hoechst labels DNA which is present in the nucleus while phalloidin labels actin which is present in the cytoplasm. Usually the analysis of these images requires to solve two types of problems: 1) the nuclei and cell segmentation, 2) the accurate cell count.

In Figure 5.7 we can see the segmentation results for the Hoechst stain. This image set has a big number of cells with similar shape and size that can be found concentrated in distinguished areas of the image. For the similarity the value can be from 0 (for no similarity at all) to 1 (for identical similarity), which means that the higher the lines are in the plot, the better for their performance. Therefore, it is observable at the plot that the design centering method is better than CMA-ES and the manual tuning in almost every frame. In table 5.3 we can also see the average values for the 12 frames. Design centering manages to improve manual tuning by 12.4% and CMA-ES by 3.59%.

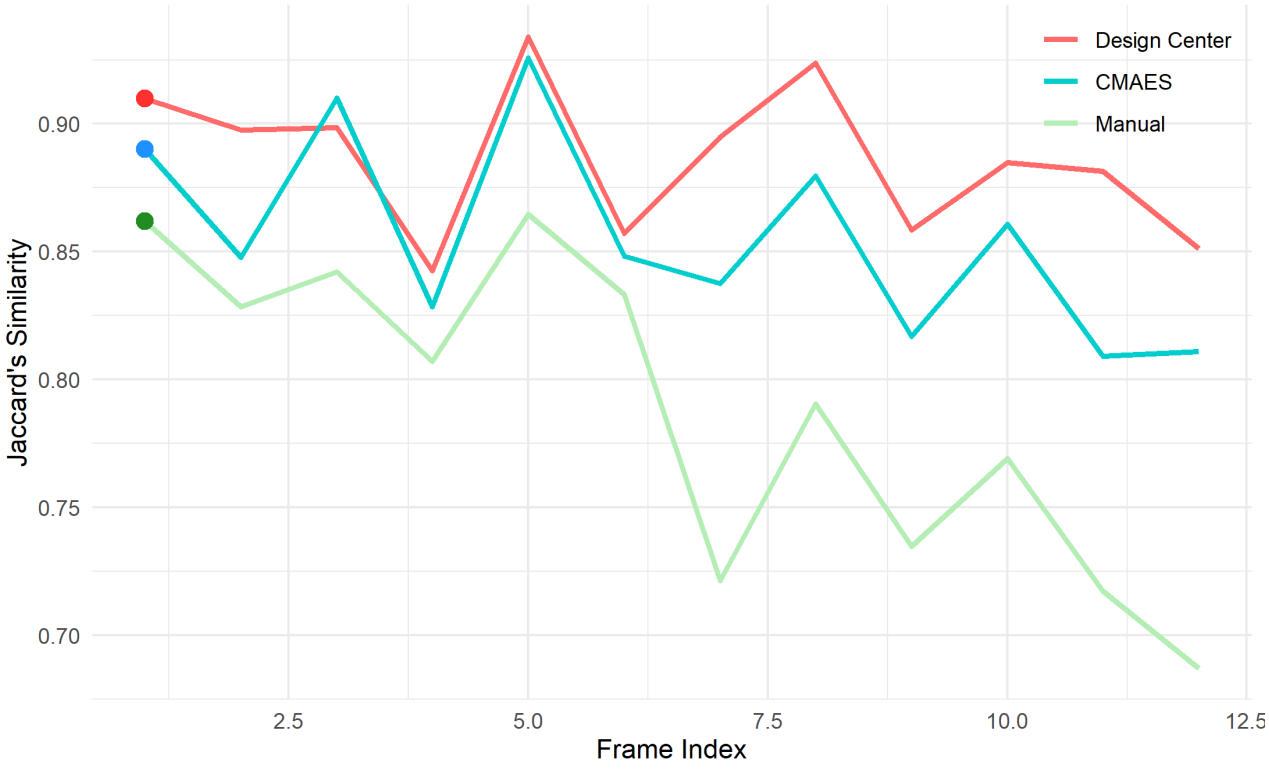
The segmentation results for the phalloidin stain can be seen in Figure 5.8. These images seem much more complex and different from the previous ones. The shape, size and brightness of the objects vary a lot, and even by eye vision it is difficult to recognize their boundaries. This time, the CMA-ES method seems to be very close to the design centering one, and even slightly outperforming it in some frames except the 10th, where design centering has a much higher similarity. Nevertheless, their big difference in this frame contributes in getting almost the same average improvement of 3.72%, as in the previous stain. Concerning the manual tuning, by the plot it seems that it has a quite low performance, in frame 3 and 8 it goes even beyond the similarity of 0.25. In general the performance of every tuning method for the phalloidin stain is much lower than the Hoechst one, as it is possible to see by comparing the two rows in table 5.3. This is probably because of the complexity of the images in this set, that makes it hard for the Squassh algorithm to perform the segmentation task.

Label	Average Jaccard's Similarity			Average Design Centering Difference	
	Manual	CMAES	Design Center	from Manual	from CMAES
humanht29bbbc008stain1	0.7882	0.8555	0.8862	+12.44%	+3.59%
humanht29bbbc008stain3	0.4138	0.5781	0.5997	+44.92%	+3.72%

TABLE 5.3: The performance results for the "Human HT29 colon-cancer cells" BBBC008 image set. The first row shows the images stained with Hoechst and the second row the ones with phalloidin.

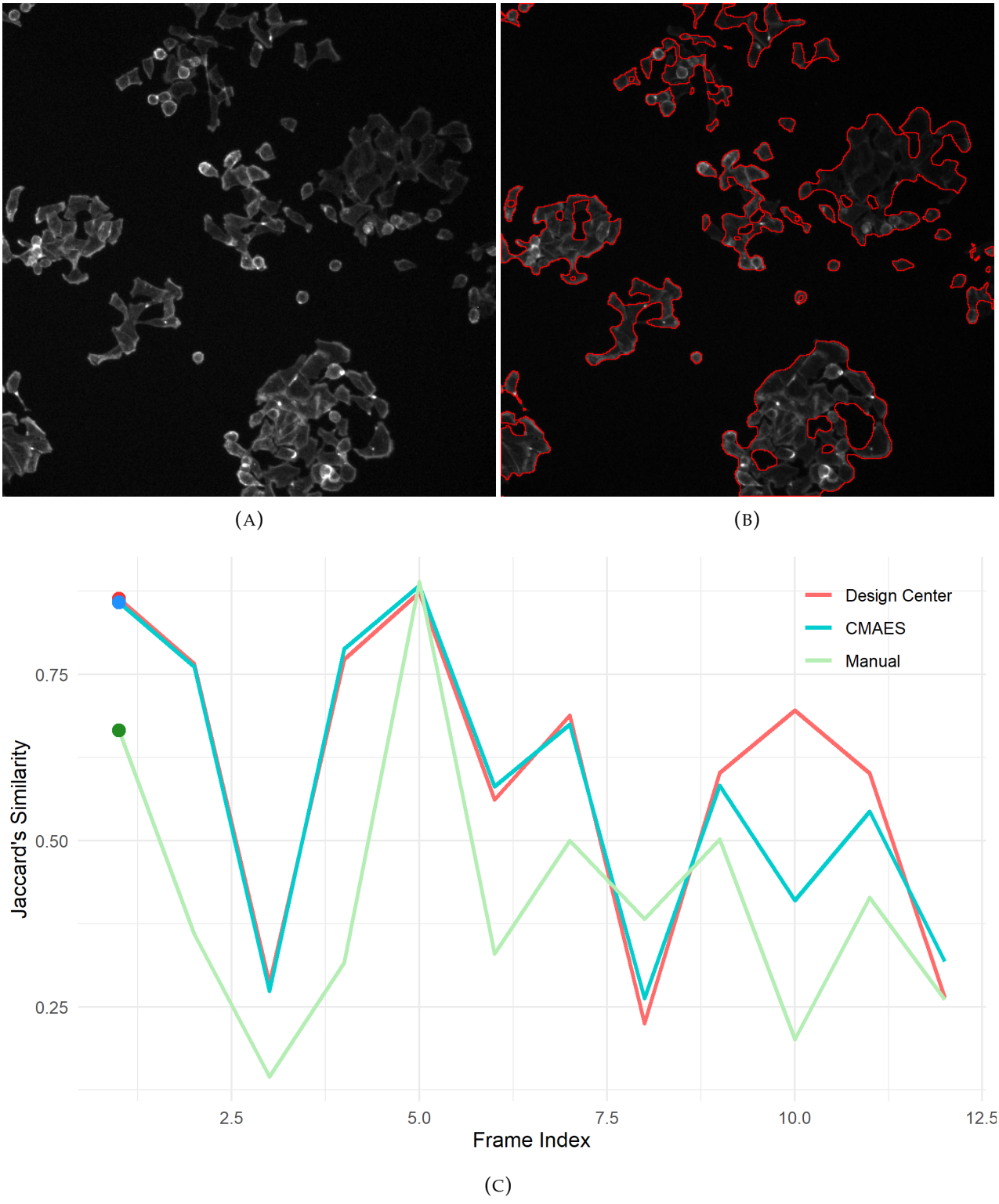


(A) (B)



(C)

FIGURE 5.7: The "Human HT29 colon-cancer cells" BBBC008 image set stained with Hoechst. (A) The first frame of the sample. (B) The result of the design centering segmentation of the first frame of the sample. (C) The plot of the similarity results for each frame of the sample.



(A) (B)

(C)

FIGURE 5.8: The "Human HT29 colon-cancer cells" BBBC008 image set stained with phalloidin. (A) The first frame of the sample. (B) The result of the design centering segmentation of the first frame of the sample. (C) The plot of the similarity results for each frame of the sample.

5.1.2 Variations in Time

The second kind of data variations that we wanted to test, are the variations in time. In this case, biological data come in the form of time-series (or videos) of a recorded experiment. Again we are using an image set from the Broad Bioimage Benchmark collection which is called "Simulated HL60 cells" with accession number BBBC035 [38]. These are synthetic images from the Cell Tracking Challenge. The images depict simulated nuclei of HL60 cells stained with Hoechst. The image sequence that was used consists of 150 frames. The number of clustered nuclei increases with time, adding more complexity to the problem. Again, the ground truth comes to a foreground/background form so we used the Jaccard similarity for comparison of the segmentation masks.

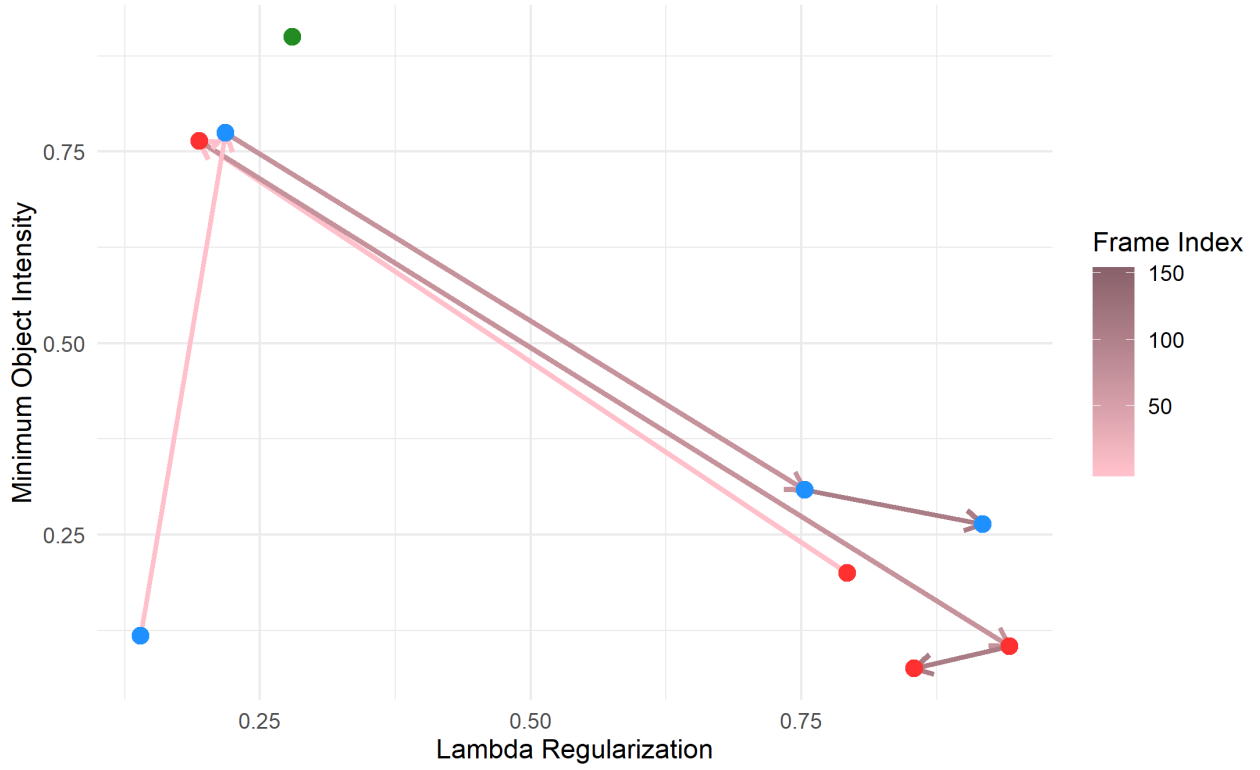
In this case the manual tuning was done for the first frame of the sequence while the CMA-ES and design centering tuning were done for 4 key-frames (1,72,108,150) as it can be seen in Figure 5.9 (B). The linear interpolation of the points found can be seen in (A) of the same figure. The dots in red show the design centers, the dots in blue show the start points from CMA-ES and the green dot shows the manual tuning parameters. It is interesting to observe that the parameters vary a lot from key-frame to key-frame. Concerning the performance, it is evident by the similarity plot that the manual tuning had the worst performance, while CMA-ES and design centering are very close. The average similarity for all the 150 frames can be seen in table 5.4. Indeed, between CMA-ES and design centering there is only a slight improvement of 0.14%.

Some frames from the segmentation results can be seen in Figure 5.10. On the left there is the manual tuning and on the right the design centering one. It is interesting to observe that both segmentations look almost the same in the first frame, but as the time passes and the cell number is increasing, manual tuning fails in segmenting many of them. This is a good example of how segmentation parameters that work very nice with one image, can have bad results when used for the whole set, due to the variation.

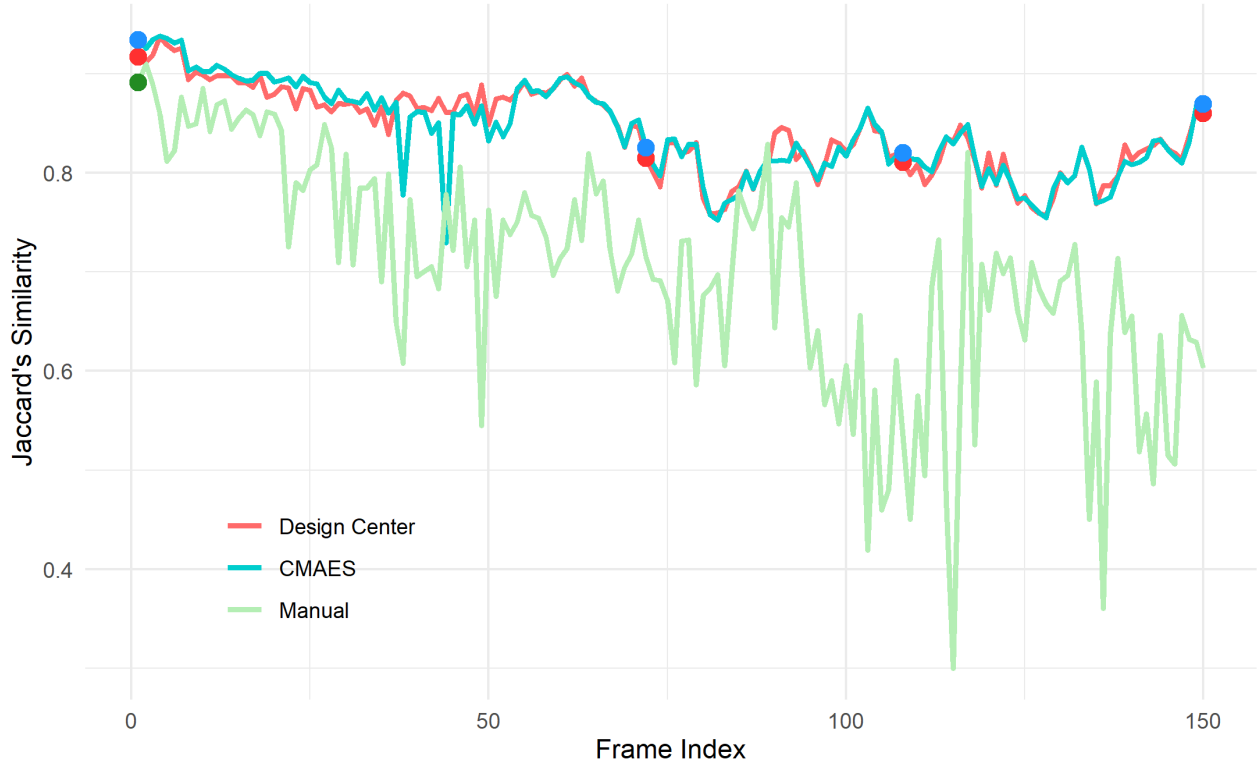
Keeping in mind that CMA-ES and design centering were so close, we thought to try to increase the data variation by using only one key-frame instead of four. The results of this test can be seen in Figure 5.11 and also in the second row of table 5.4. As expected, by increasing the data variation design centering is improving CMA-ES by 1.4%, which even though is not really high, it is ten times higher than the improvement of the previous test with 4 key-frames. One other thing to observe in this plot is that similarity on the first frame, which is the training frame, is higher for CMA-ES. This makes sense as CMA-ES is an optimization algorithm and so in theory it is supposed to find better segmentation parameters for this single frame.

k	Average Jaccard's Similarity			Average Design Centering Difference	
	Manual	CMAES	Design Center	from Manual	from CMAES
4	0.6995	0.8416	0.8428	+20.49%	+0.14%
1		0.8269	0.8385	+19.86%	+1.40%

TABLE 5.4: The performance results for the "Simulated HL60 cells" BBBC035 image set. The first row shows the test with 4 key-frames used and the second row the test with one key-frame.

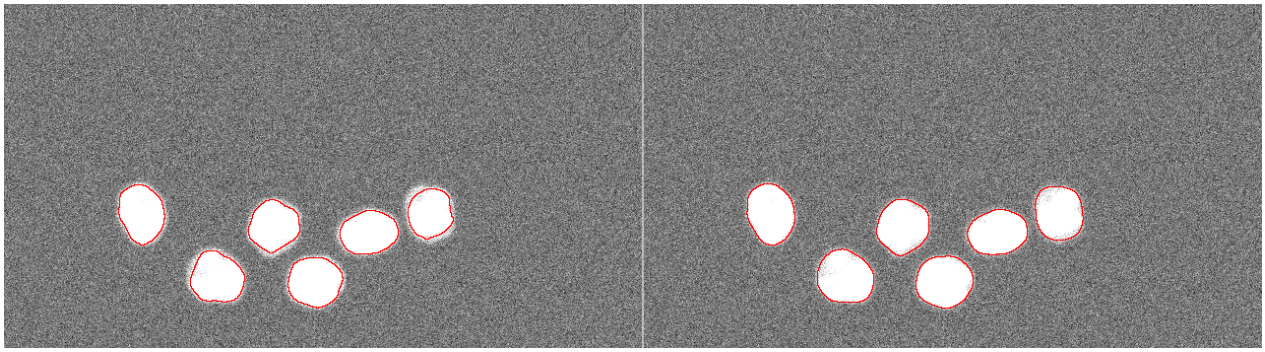


(A)

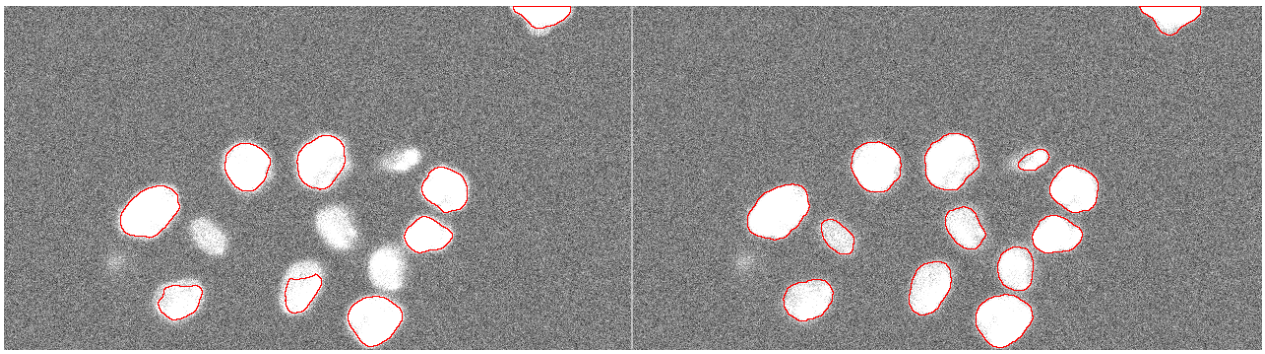


(B)

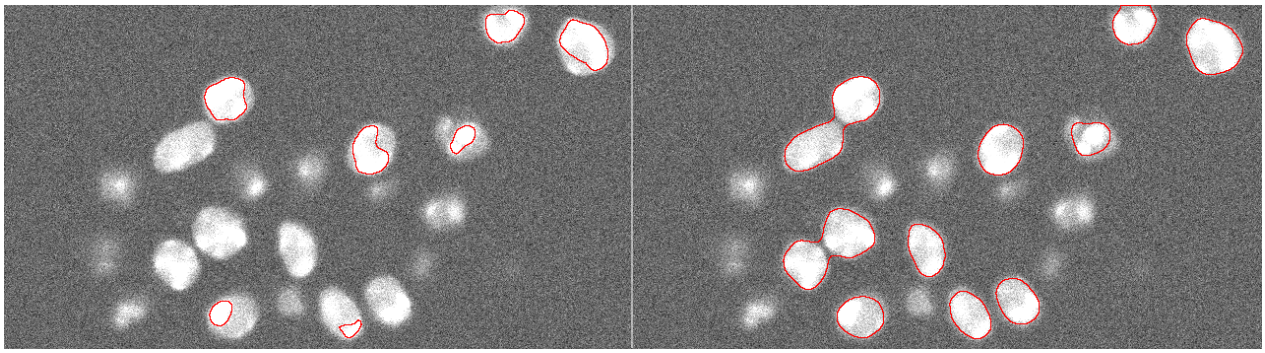
FIGURE 5.9: Results for the "Simulated HL60 cells" BBBC035 time-series. (A) The plot of the key-frame parameters and their linear interpolation. The red dots are for the design centers. The blue dots are for the CMA-ES start points. The green dot is for the manual tuning parameter set. (B) The plot of the similarity results for each frame of the image set, with the use of 4 key-frames (represented by dots with the corresponding color for each method).



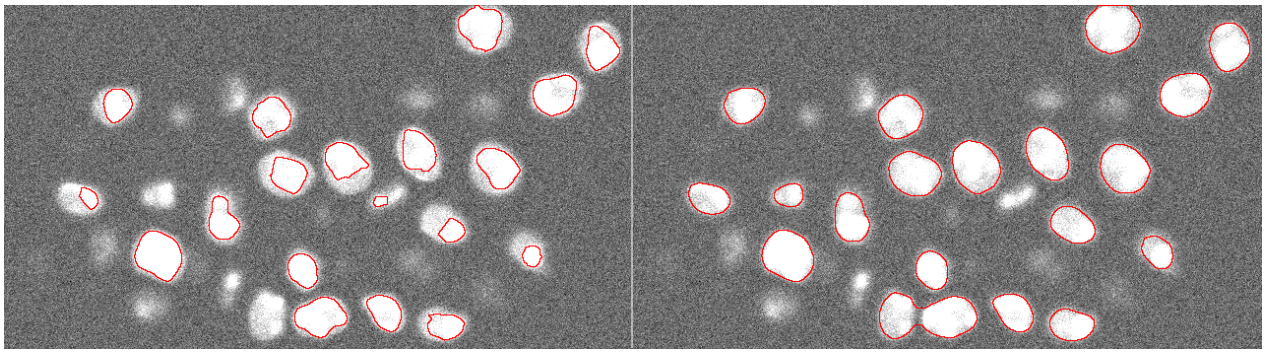
(A) Frame: 4



(B) Frame: 68



(C) Frame: 115



(D) Frame: 150

FIGURE 5.10: Different frames from the "Simulated HL60 cells" image set with their segmentation outlines layer. The images on the left show the results from the manual parameter tuning method while the images on the right show the results from the automatic design centering tuning.

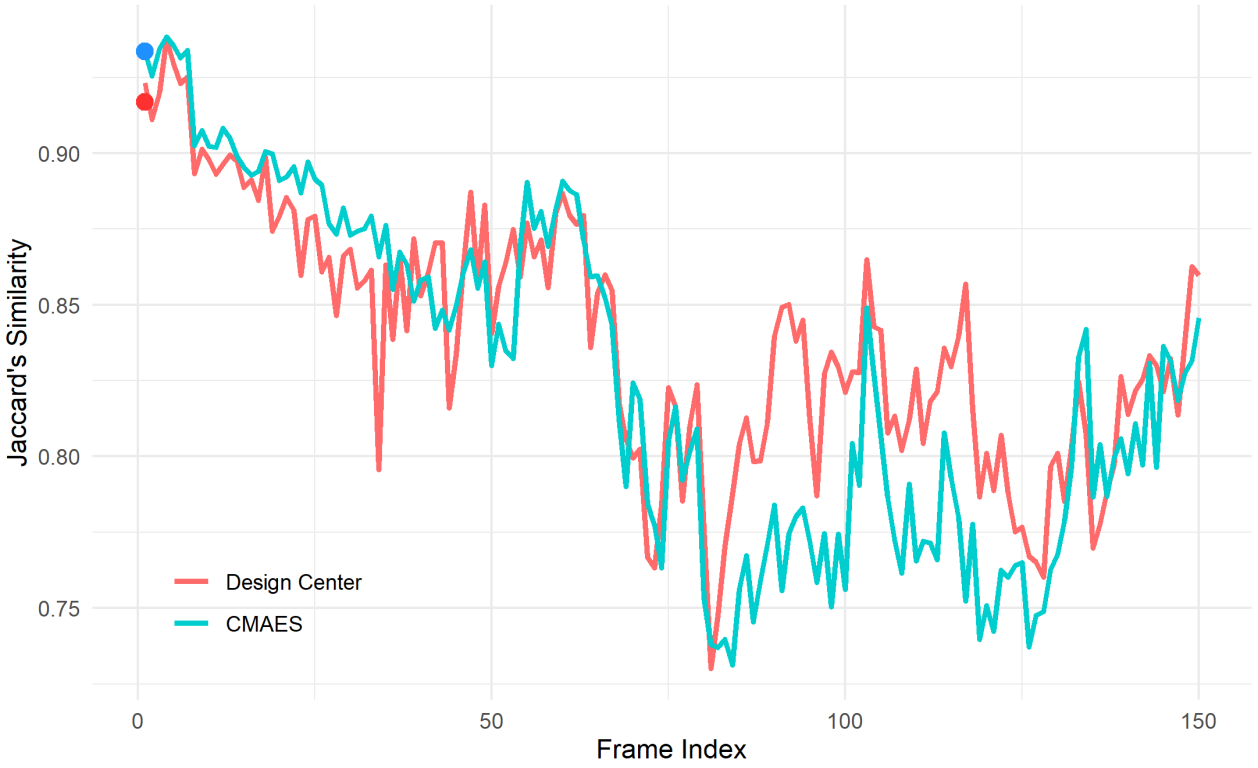


FIGURE 5.11: The plot of the similarity results for each frame of the "Simulated HL60 cells" image set, with the use of 1 key-frame (represented by the dots on frame 1 with the corresponding color for each method).

5.1.3 Variations in Staining

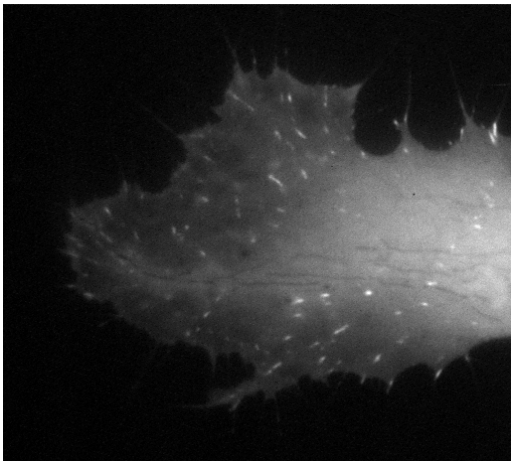
In this section we present the results from the test of robustness against the variations in staining. The image set used comes from the Cell Image Library with CIL:42607 [39]. It contains three videos of a goldfish (CAR) fish fibroblast taken with confocal microscope and showing the dynamics of the focal adhesion protein paxillin (stain 1), microtubule tip protein EB3 (stain 2), and actin (stain 3). The idea of this test is to do the training for the first stain and try the same parameters for the other stains. The goal is to segment the cell outline. For the testing we used 5 frames from each time-series. The ground truth in this case was produced by us as a foreground/background for all the frames.

The segmentation similarity to ground truth results can be seen in Figure 5.12. The training was done for the first frame of the image set (B), that is why there are no dots in (D) and (F). For the first stain, since we have training and evaluation on the same image set, the variation that we have is only in time. In this case, design centering had worse performance than both the manual tuning and CMA-ES (table 5.5). Going now to the stain variations, the results were mixed. For the second stain, design centering outperformed CMA-ES and manual tuning, but in the third stain manual tuning had a slightly better performance. Consequently, it is difficult to say if design centering managed to give more robust solutions for this image set. The mixed results could be because there is too much variation, as different stains look really different, or because of low number of oracle evaluations.

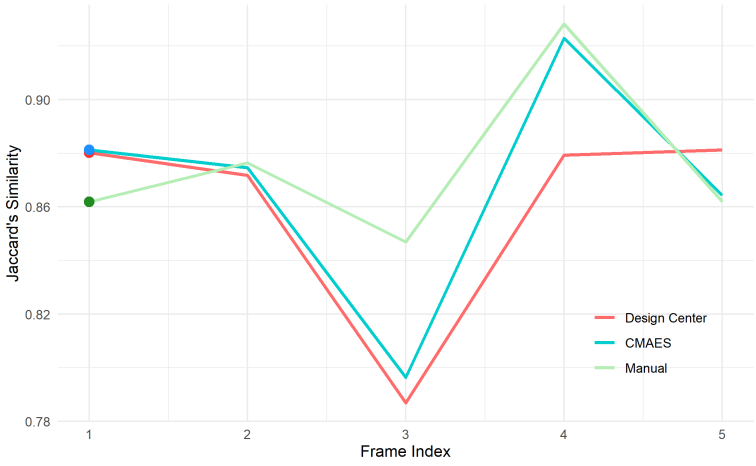
Of course, there is always the scenario that there was not enough variation in the data, but since we cannot quantify the variation we cannot be sure if that is the reason for the mixed results.

Stain	Average Jaccard's Similarity			Average Design Centering Difference	
	Manual	CMAES	Design Center	from Manual	from CMAES
stain1	0.8751	0.8679	0.8599	-1.737%	-0.924%
stain2	0.8915	0.8913	0.8925	+0.103%	+0.130%
stain3	0.8687	0.8655	0.8653	-0.361%	+0.0304%

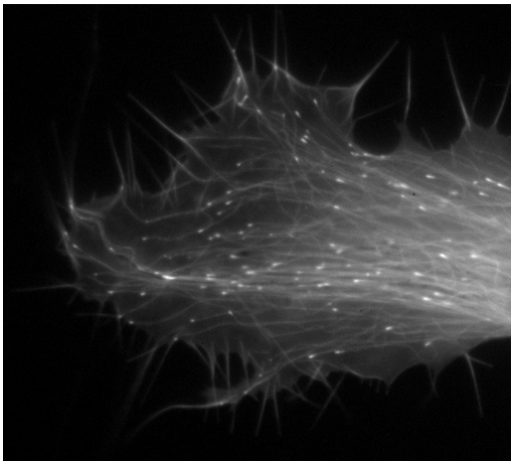
TABLE 5.5: The performance results for the CIL:42607 image set from the Cell Image Library.



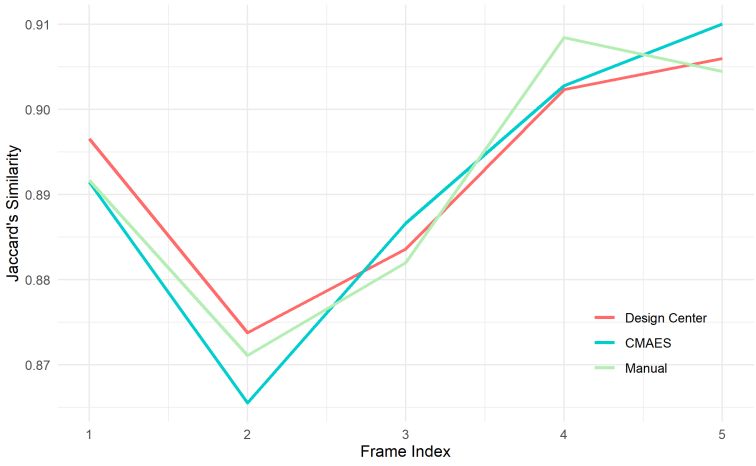
(A)



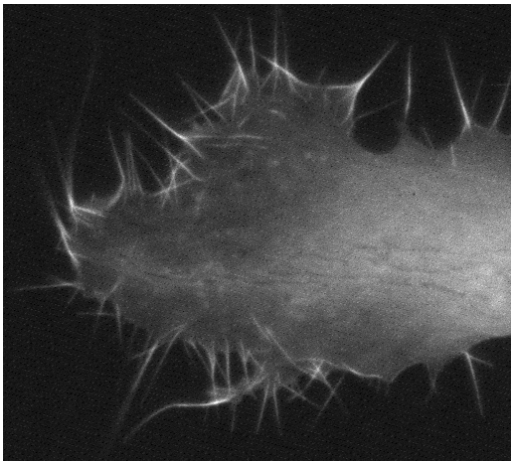
(B)



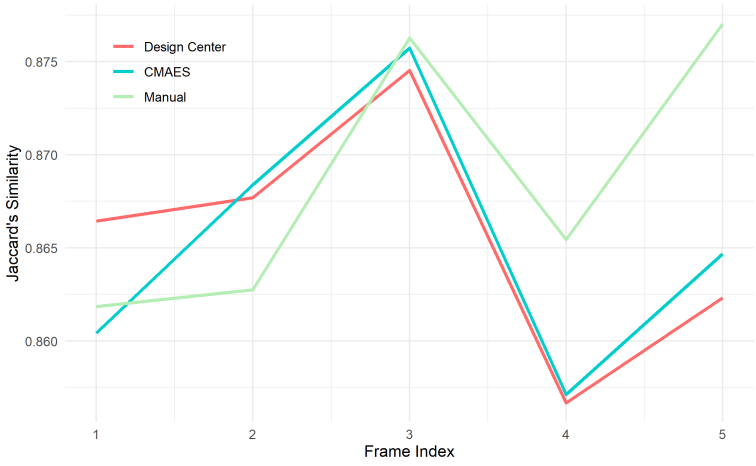
(C)



(D)



(E)



(F)

FIGURE 5.12: The CIL:42607 image set from the Cell Image Library, showing a goldfish (CAR) fish fibroblast with three different stains (A),(B) Image of the first frame and segmentation similarity plot for stain 1. (C),(D) Image of the first frame and segmentation similarity plot for stain 2. (E),(F) Image of the first frame and segmentation similarity plot for stain 3.

5.1.4 Variations in Imaging Depth

The last kind of data variations that we want to test, are variations in imaging depth. In this case, biological data come in the form of a z-stack which is a result of three-dimensional imaging. Again we are using an image set from the Broad Bioimage Benchmark collection which is called "3D HL60 Cell line" with accession number BBBC024vl [40]. This dataset contains synthetic images with 20 Hl60 cell nuclei and 98 z-planes. The ground truth again is a foreground/background segmentation.

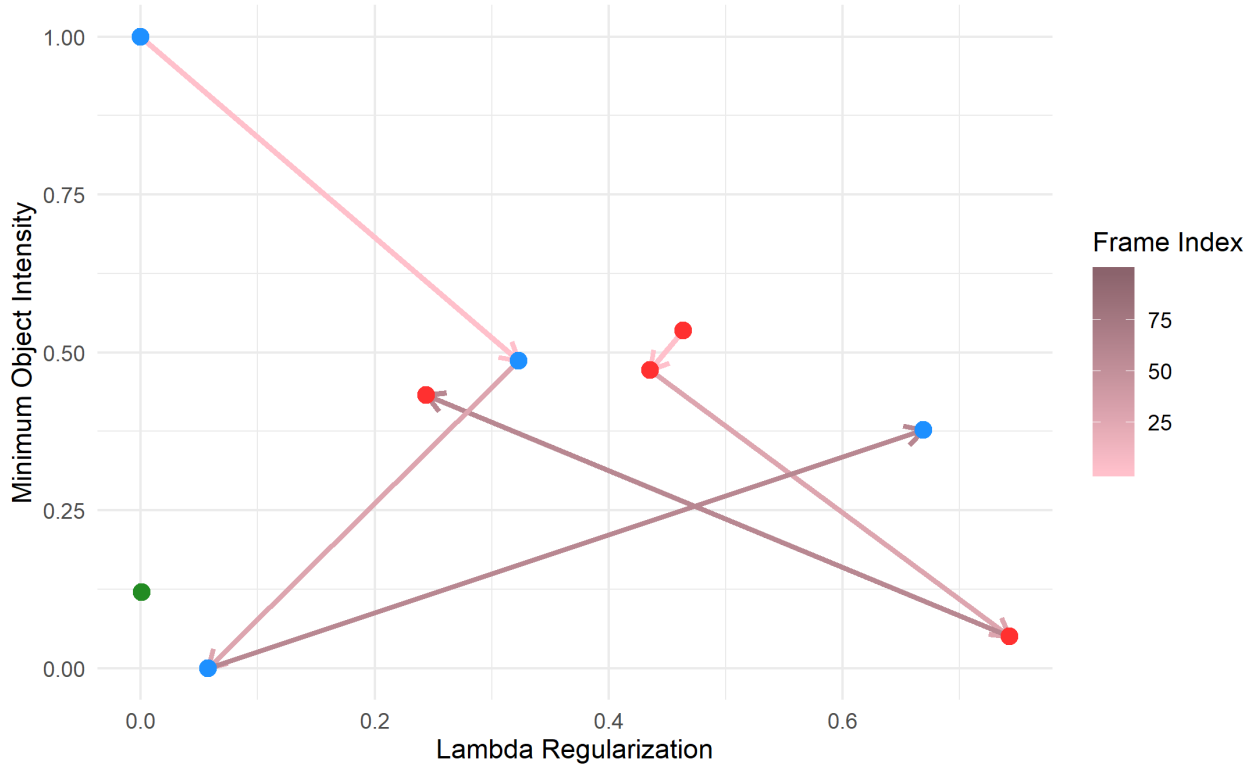
For this case, the manual tuning was done for the first frame of the sequence while the CMA-ES and design centering tuning were done for 4 key-frames (1,28,59,98) as it can be seen in Figure 5.13 (B). The linear interpolation of the points found can be seen in (A) of the same figure. Again, we can observe that the parameters vary a lot from key-frame to key-frame. An interesting area in this plot is the part of the first 15 frames where we can see that the similarity values are either 0 or 1. This happens because the ground truth for these frames has no segmented objects, even though there are some blurry objects (Figure 5.14 (A)), and so if Squassh segments something the similarity becomes 0 (Figure 5.14 (B) upper image). In the same part we can notice that the design centering is mostly 0. This happened because the feasible region has a doughnut form (because only the parameter limits give an empty segmentation) and so the design center exists in the hole of the doughnut, which is outside the feasible region. This can be confirmed by seeing the first point taken for CMA-ES (blue) and the design centering (red) in Figure 5.13 (A). Concerning the performance, the manual parameter tuning had the worst performance, while CMA-ES and design centering are quite close for most of the frames after the 15th. Due to these first frames, CMA-ES had a better average performance, as we can see in table 5.6.

Some frames from the segmentation results can be seen in Figure 5.14. The frames above show the design centering tuning and the frames below the CMA-ES. It is interesting to observe that there are some frames where CMA-ES segments noise, for example frame 55 (Figure 5.14 (D)) which can be also noticed as a sudden decrease in the similarity plot. In general, CMA-ES seems to have an overfitted segmentation around this frame, which is probably because of the low lambda regularization for the third key-frame.

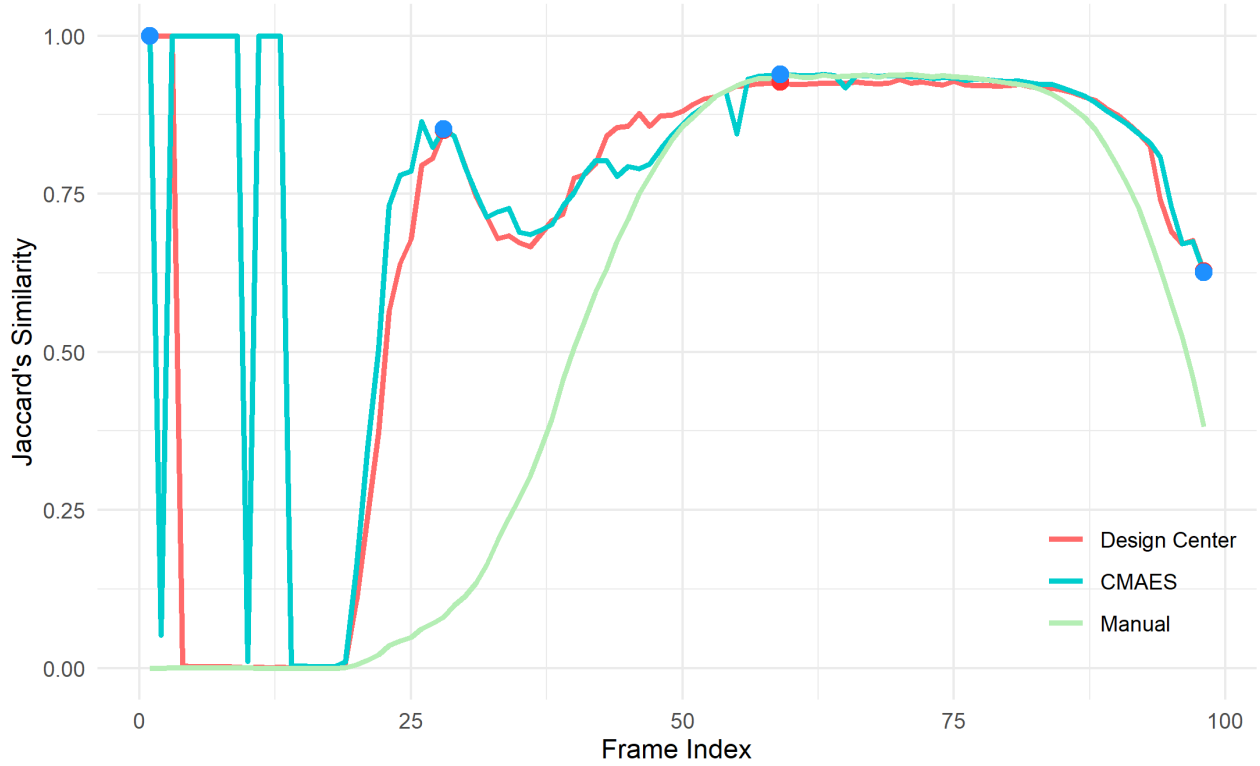
We also tried to increase the data variation by using only one key-frame instead of four. The results of this test can be seen in Figure 5.15 and also in the second row of table 5.6. The result is that both CMA-ES and design center have a lower average similarity than before, but this time design center is improving over CMA-ES by 8.26%.

k	Average Jaccard's Similarity			Average Design Centering Difference	
	Manual	CMAES	Design Center	from Manual	from CMAES
4	0.532	0.782	0.692	+30.07%	-11.48%
1		0.533	0.577	+8.44%	+8.26%

TABLE 5.6: The performance results for the "3D HL60 Cell line" BBBC024vl image set. The first row shows the test with 4 key-frames used and the second row the test with one key-frame.

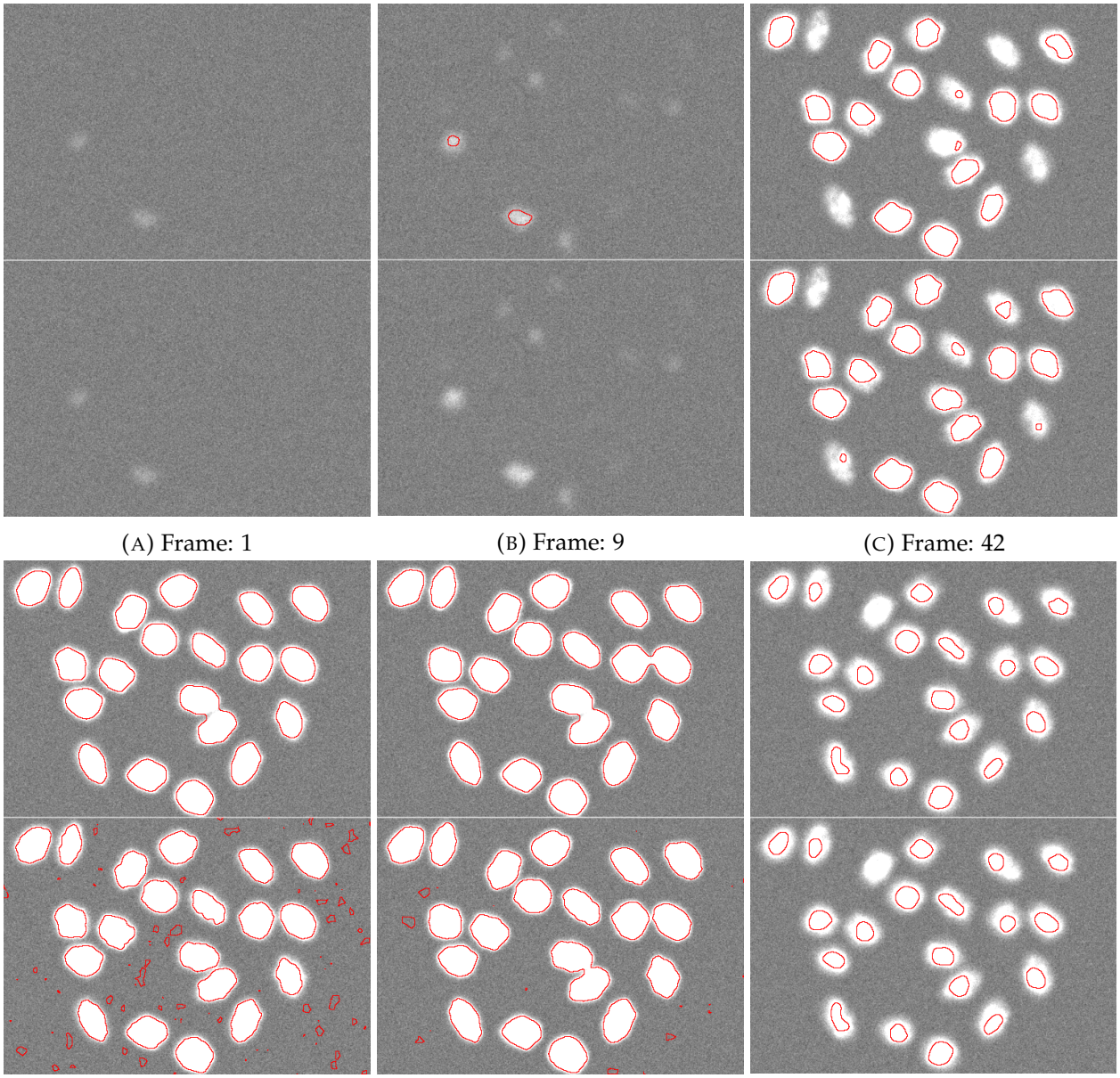


(A)



(B)

FIGURE 5.13: Results for the "3D HL60 Cell line" BBBC024v1 z-stack. (A) The plot of the key-frame parameters and their linear interpolation. The red dots are for the design centers. The blue dots are for the CMA-ES start points. The green dot is for the manual tuning parameter set. (B) The plot of the similarity results for each frame of the image set, with the use of 4 key-frames (represented by dots with the corresponding color for each method).



(A) Frame: 1 (B) Frame: 9 (C) Frame: 42

(D) Frame: 55 (E) Frame: 65 (F) Frame: 98

FIGURE 5.14: Different frames from the "3D HL60 Cell line" image set with their segmentation outlines. The images on top show the results from the manual parameter tuning method while the images on bottom show the results from the automatic design centering tuning.

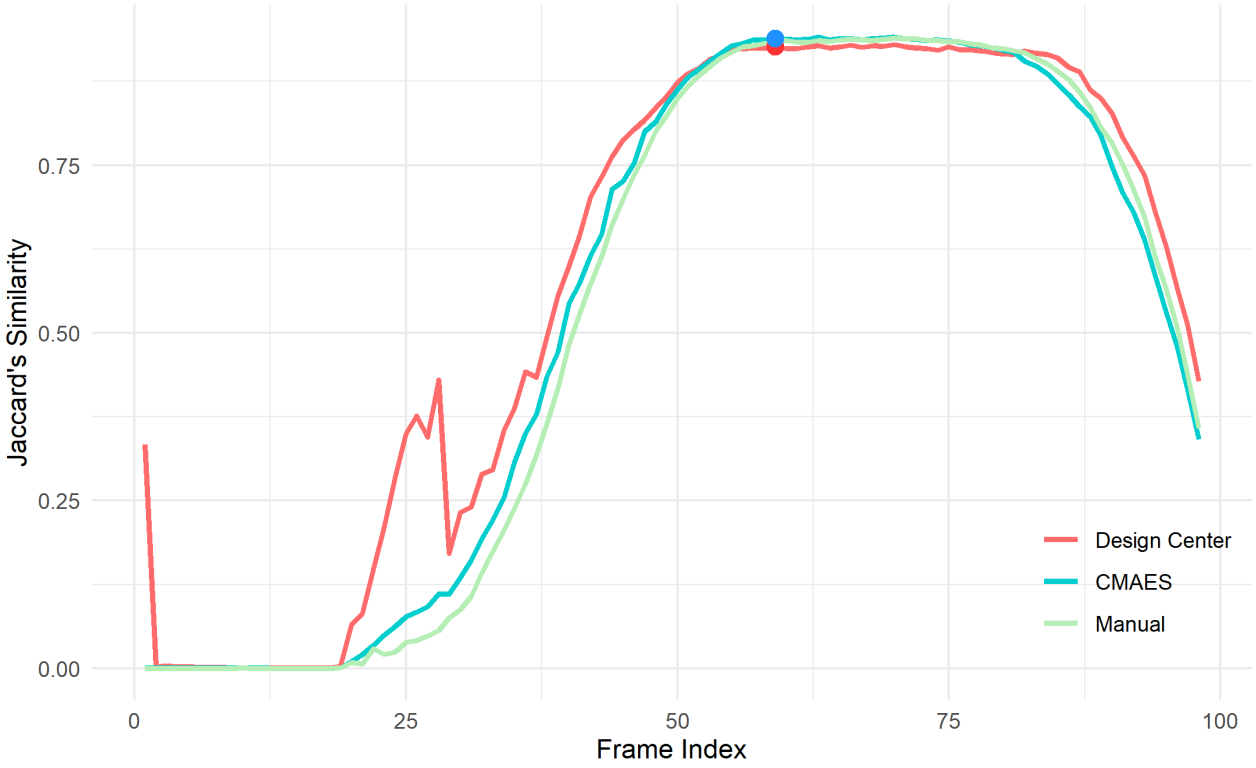


FIGURE 5.15: The plot of the similarity results for each frame of the "3D HL60 Cell line" image set, with the use of 1 key-frame (represented by dots with the corresponding color for each method).

5.2 Benchmark Analyses

5.2.1 Quality benchmarks based on the number of evaluations

As we saw previously, the number of evaluations used for the tests above was 300. This is because by experience this number seemed to be enough in most cases for the Lp-Adaptation to find a good design center. However, we thought that it would be also valuable information to find out how the Lp-Adaptation behaves for different numbers of evaluations. So we run Lp-Adaptation for the "Human HT29 colon-cancer cells" BBBC001 image set for for different number of evaluations in a range from 5 to 1000. The results can be seen in Figure 5.16. The y-axis is the deviation from the ground truth, which means that lower values have better performance. Each point represents an individual run of Lp-Adaptation. We can see that from 0 to 125 evaluations, the design center is the same and does not have so good performance. Probably this is because most of the points that are sampled are not in the feasible region and so the design center is not updated, only the sampling radius changes depending on the hitting probability. As the evaluations increase, Lp-Adaptation finds better parameters quite fast and the deviation seems to fluctuate around a stable value. For 500 evaluations it has the lowest deviation, which seems to go up again as the evaluations increase. Since we are using an increasing hitting probability, this could happen due to overfitting.

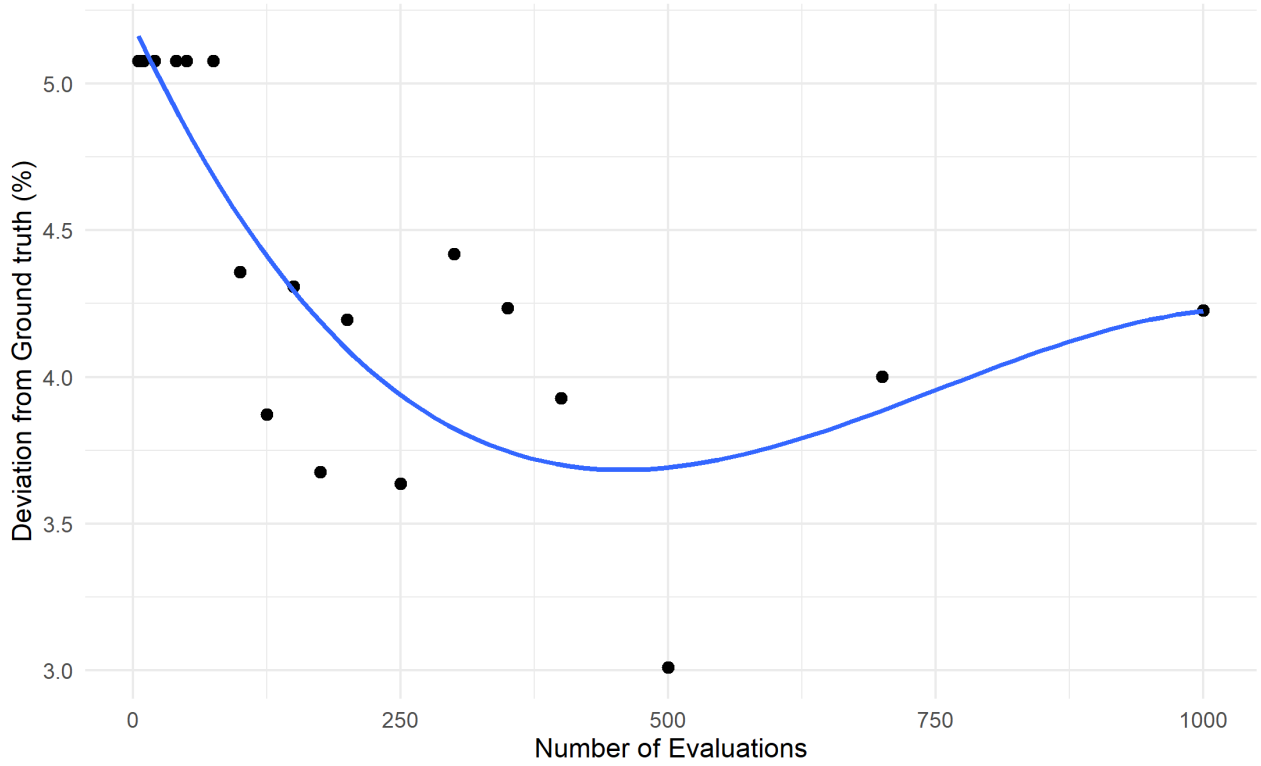


FIGURE 5.16: Performance on different number of evaluations.

5.2.2 Time Benchmarks

In this section we are going to present some time benchmarks of the the automatic segmentation process. Time plays an important role in the overall efficiency of the plugin and it is also a significant factor for the users. In table 5.7 we can see the time measurements from the 14 tests that we performed to get the results of the previous section. Concerning the symbols of the table: n_f is the number of frames, k is the number of key-frames, n_{CMA} is the maximum number of evaluations for CMA-ES and n_{DC} is the number of evaluations for Lp-Adaptation. The overall time for the tests varies from 41 minutes up to 9 hours and 58 minutes. This variation in time happens because of differences either in the data or the algorithmic options used. Concerning the data, two important parameters are the size and the number of frames that they include. The size affects the Squassh segmentation time in all the stages of the process while the number of frames only affects the last step of computing the final result. For the algorithmic options, important factors are the number of key-frames, the maximum number of evaluations for CMA-ES and the the number of evaluations for Lp-Adaptation. The number of key-frames basically shows us how many design centers are computed and so its increase multiplies the tuning time. The evaluation numbers show us how many times we run a single segmentation. Overall, the number of all segmentation runs can be calculated like this:

$$n = k(n_{CMA} + n_{DC}) + n_f \tag{5.1}$$

where n is the number of segmentation runs.

Input Data			Algorithmic Options			Time			
Label	n_f	Size(MB)	k	n_{CMA}	n_{DC}	CMA-ES(min)	Design Center(min)	Overall(min)	Overall/n (s)
droskc167s1	10	2.5	1	120	300	6.7	33.1	41.3	5.8
droskc167s2	10	2.5	1	120	300	8.0	41.0	51.2	7.1
droskc167s3	10	2.5	1	120	300	11.4	38.3	52.2	7.3
droskc167s4	10	2.5	1	120	300	17.2	46.5	66.4	9.3
droskc167s5	10	2.5	1	120	300	9.7	52.3	65.0	9.1
humanht29bbbc001	6	1.5	1	120	300	11.7	71.1	86.9	12.2
humanht29bbbc008stain1	6	3.0	1	120	300	37.5	50.5	90.6	12.8
humanht29bbbc008stain3	6	3.3	1	120	300	44.5	54.1	100.9	14.2
3dhl60	98	32.8	4	120	300	221.3	334.8	596.7	20.1
simhl60	150	31.9	4	120	300	167.4	251.8	455.6	14.9
ncbigold	5	1.34	1	120	300	33.75	74.6	111.1	15.7
mcbalb3t3	10	2.5	1	50	200	21.5	24.9	49.7	11.5
mfc10afib	30	9.86	1	50	200	16.6	24.6	46.0	9.9
helaklmna	31	5.32	3	50	200	27.1	41.3	73.49	5.7

TABLE 5.7: Time benchmarks for the automatic tuning segmentation of different images.

Now, the time for a single segmentation of an image is not fixed for Squassh as it depends on the parameters. From experience we found that for Squassh, the lower the parameters are the more time it takes. But one question that comes up from seeing these time measurements is what part of the design centering estimation process is because of Squassh. In order to find that we run again Squassh for all the sampled points of three of the tests and found the results that can be seen in table 5.8. We can see that the percentage of the Squassh time in the whole process varies from 85%-98%. This means that Squassh is the bottleneck of our method and so the time performance of the segmentation algorithm affects significantly the running time of the tuning process.

Label	Squassh Time(min)	Design Center(min)	Percentage of Squassh
droskc167s1	32.5	33.1	98%
humanht29bbbc001	61.6	71.1	87%
humanht29bbbc008stain1	43.0	50.5	85%

TABLE 5.8: Comparing the time of Squassh to the time of the whole Lp-Adaptation run.

5.3 Usability Study

5.3.1 Process

In the context of testing the Apeba segmentation plugin we also wanted to get some feedback on its usability and the opinion of real users about its efficiency and difference from the manual tuning process. That's why we organised a usability test. The steps of the test were the following:

1. Pre-Questionnaire.
2. Manual parameter tuning for three image sets using the original Squassh Fiji plugin.
3. Automatic parameter tuning for the same three image sets using the Apeba segmentation plugin.
4. Observation of the segmentation results for both methods of tuning.
5. Post-Questionnaire.

The time given for the manual tuning was 10 minutes for each image and the time of using the Apeba plugin was always lower than 10 minutes. The first three steps took an average time of 50 minutes. After that, the results were analyzed and sent to the participants who had to subjectively evaluate them and answer the post-questionnaire. The test was done with 7 participants; all of them are researchers at MPI-CBG, coming from various scientific backgrounds.

In the following sections we are going to present and analyze the answers to the two questionnaires.

5.3.2 Pre-Questionnaire

The Pre-Questionnaire consisted of two parts. In the first part the users were asked some general background questions about their prior experiences in related tasks while the second part had some opinion questions concerning the bioimage analysis process and the parameter tuning.

In Figure 5.17 we can see the results from the background questions. The first question shows that we had participants from various levels of experience in bioimage analysis. This is good because the aim of our plugin is to be used both by beginners and expert users so every feedback was valuable. From the rest of the questions we can see that the experience of the participants in manual tuning in average was close to the middle while almost all participants had no experience with automatic parameter tuning.

In Figure 5.18 we can see the results of the opinion questions. It is interesting to observe that the participants tented to feel comfortable to use an automatic system for parameter tuning, even though, as we saw, they had no relevant experience. Also, most participants would prefer the automatic tuning over the manual one, which is an indication of reliance upon the modern AI technologies in biology.

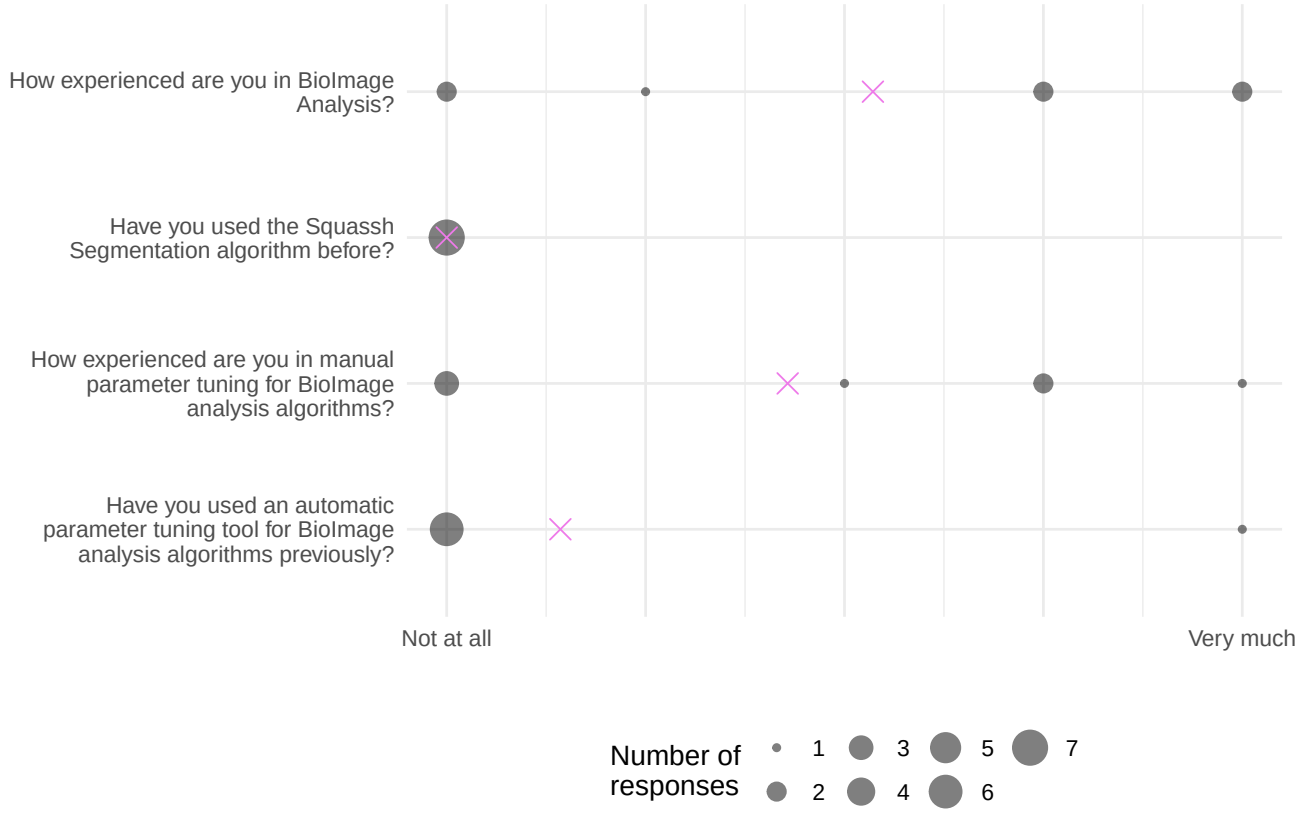


FIGURE 5.17: First part of the Pre-Questionnaire: Background Questions

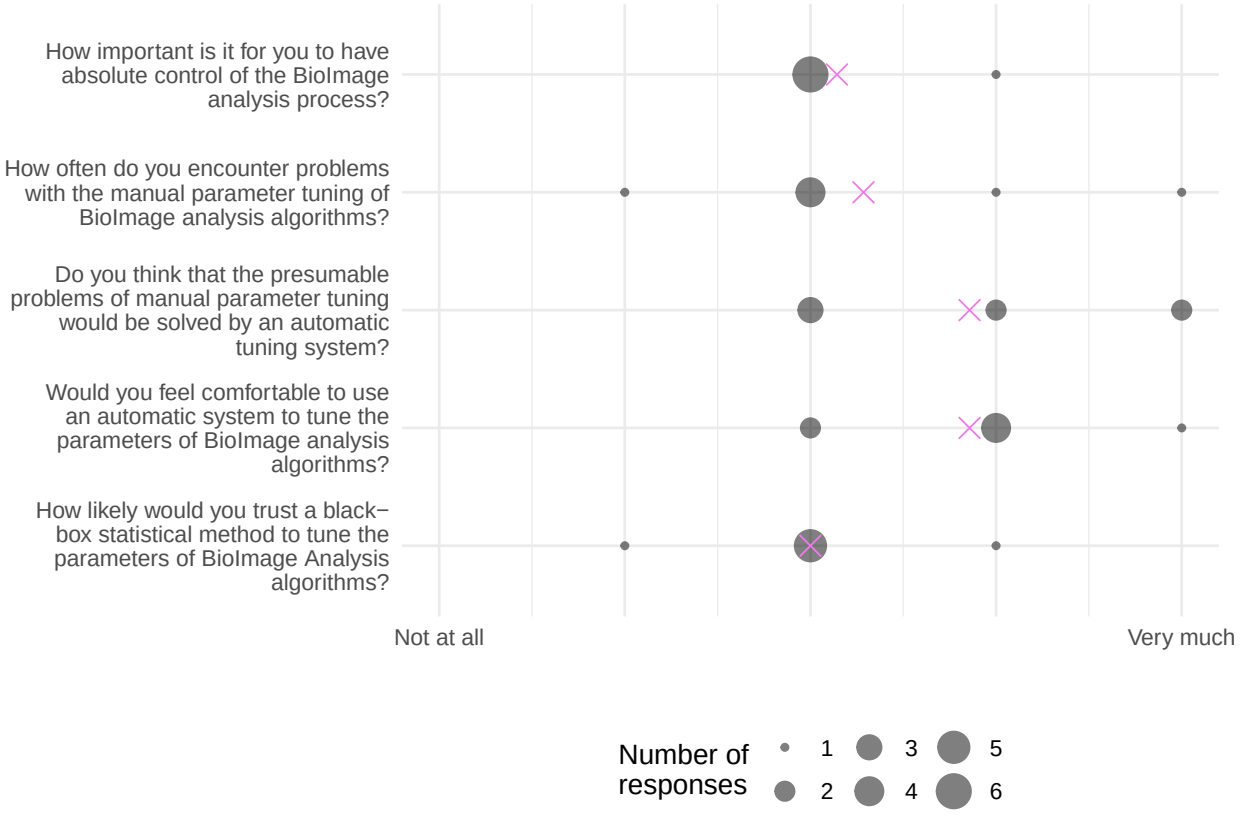


FIGURE 5.18: Second part of the Pre-Questionnaire: Opinion Questions

5.3.3 Post-Questionnaire

Usability

The first part of the Post-Questionnaire contained questions about the usability of the plugin. Most participants found the plugin easy to use and didn't feel "stuck". Also, many participants found the 'Help' tips of the plugin useful. These tips came in the form of an already filled image example for each sub-oracle (as it can be seen in Appendix A) and some tips concerning their use and effect on the final result (mostly the things that were presented in chapter 4). Finally, most participants found the plugin easy to use for users with no experience.

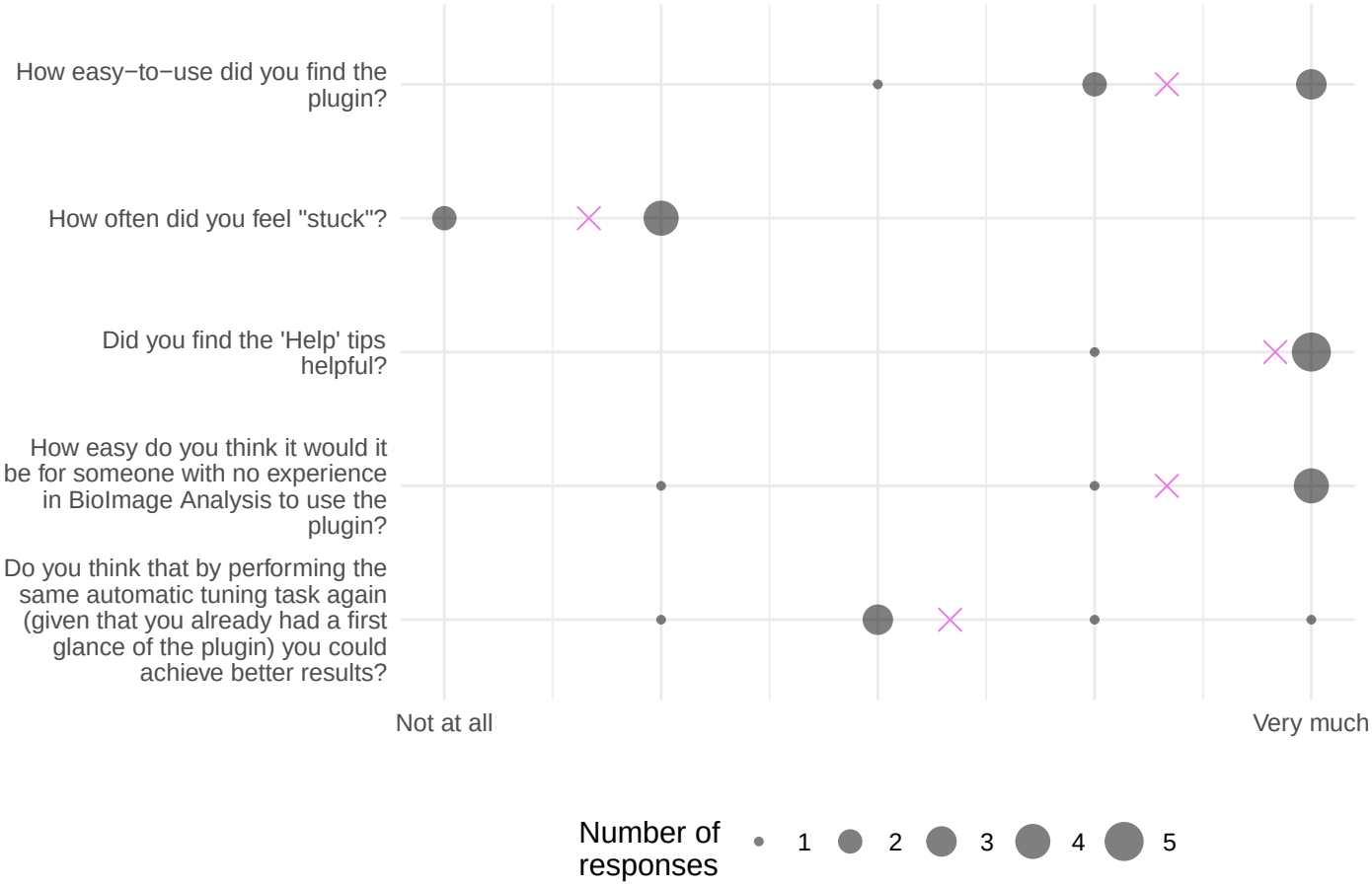


FIGURE 5.19: Usability Questions.

5.3.4 Comparing the segmentation results

In this section we are going to present the results from the users' manual and automatic tuning on three different image sets. All image sets are time-series. After seeing the segmentation results for each method, the users had to decide which method had better results.

Image Sequence 1: MC:BALB/c 3T3 Fibroblast cell

The first image set comes from the Cell Image Library with CIL:8049. The images in this time sequence show a mouse fibroblast cell line stably expressing MeCP2-GFP. We can see how the images look in Figure 5.21. The segmentation task in this case was to segment the nuclear chromatin of the 3T3 fibroblast cells. The time sequence consisted of 10 frames and the training was done on the first frame.

In Figure 5.20 we can see the answers from the participants of the usability test when asked to compare the manual with the automatic tuning method. Automatic tuning had on average slightly better results.

In Figure 5.21 (A) we can see one example from the segmentation results of the first frame, with the manual tuning on the left and the automatic one on the right. The segmentation outlines look very similar, but manual tuning had two more small objects segmented that are probably noise or out-of-focus structures. In 5.21 (B) we can see all the result images layered into one. Here it is even more clear to see the small objects that are segmented by the manual tuning, probably due to low lambda regularization.

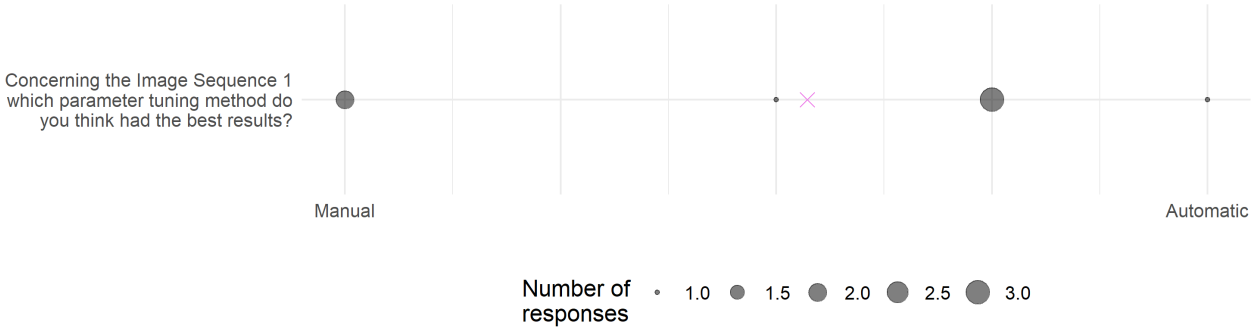
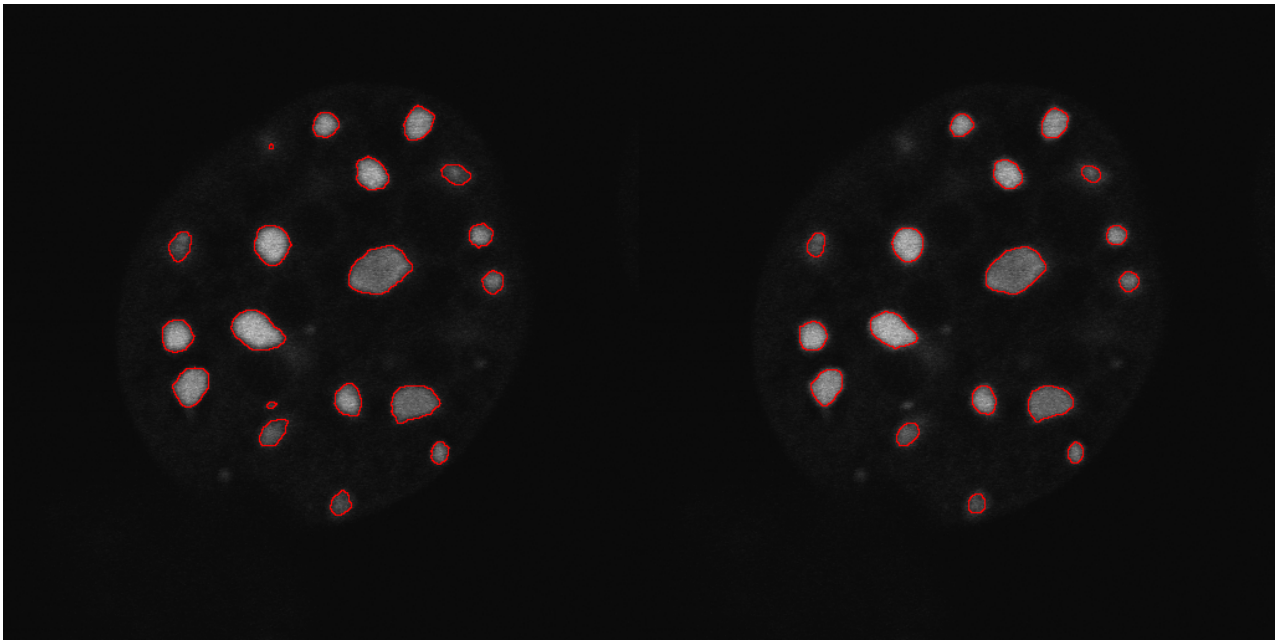
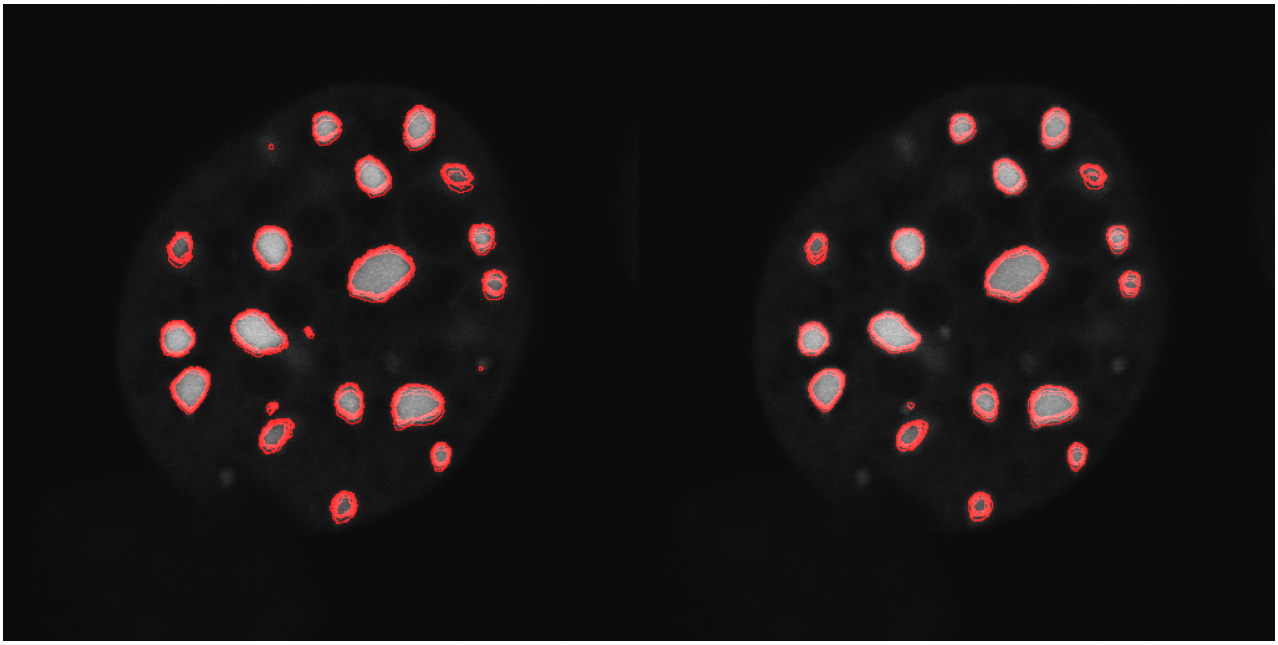


FIGURE 5.20: Comparing the segmentation results for the MC:BALB/c 3T3 Fibroblast cell



(A) Frame: 1



(B) All frames overlayed

FIGURE 5.21: Segmentation final results for the MC:BALB/c 3T3 Fibroblast with one of the bigger improvements from manual to automatic tuning, from an experienced participant in bioimage analysis. On the left there is the manual tuning whereas on the right there is the automatic one.

Image Sequence 2: Epithilial Cells MFC 10A

Again the image set was taken from the Image Cell Library with CIL 12287. In these images we can see three epithelial cells that are attached to each other and moving over time. The task in this case was to segment the whole area covered by the three cells. The time sequence consisted of 30 frames and the training was done on the first frame.

In Figure 5.22 we can see the answers from the participants of the usability test when asked to compare the manual with the automatic tuning method. In this case the automatic tuning had better results than before. One explanation to that would be that the second image set has more variation and so it is benefits more from the design centering parameters.

In Figure 5.23 we can see different frames from the image set with their segmentation outline. In the images on top we can see the manual tuning results and on the bottom the automatic tuning results. Finding a good segmentation for these images was basically to achieve the biggest area. So design centering seems to have two advantages: 1) less noisy segmentation, 2) better fit to the area covered.

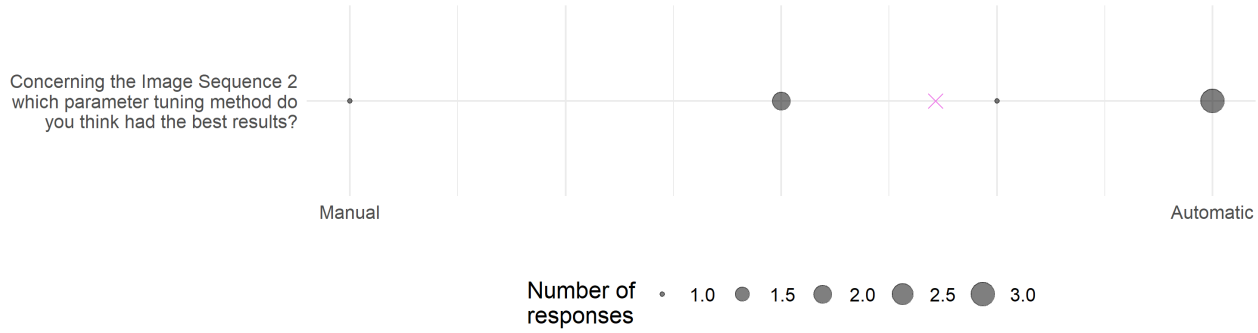
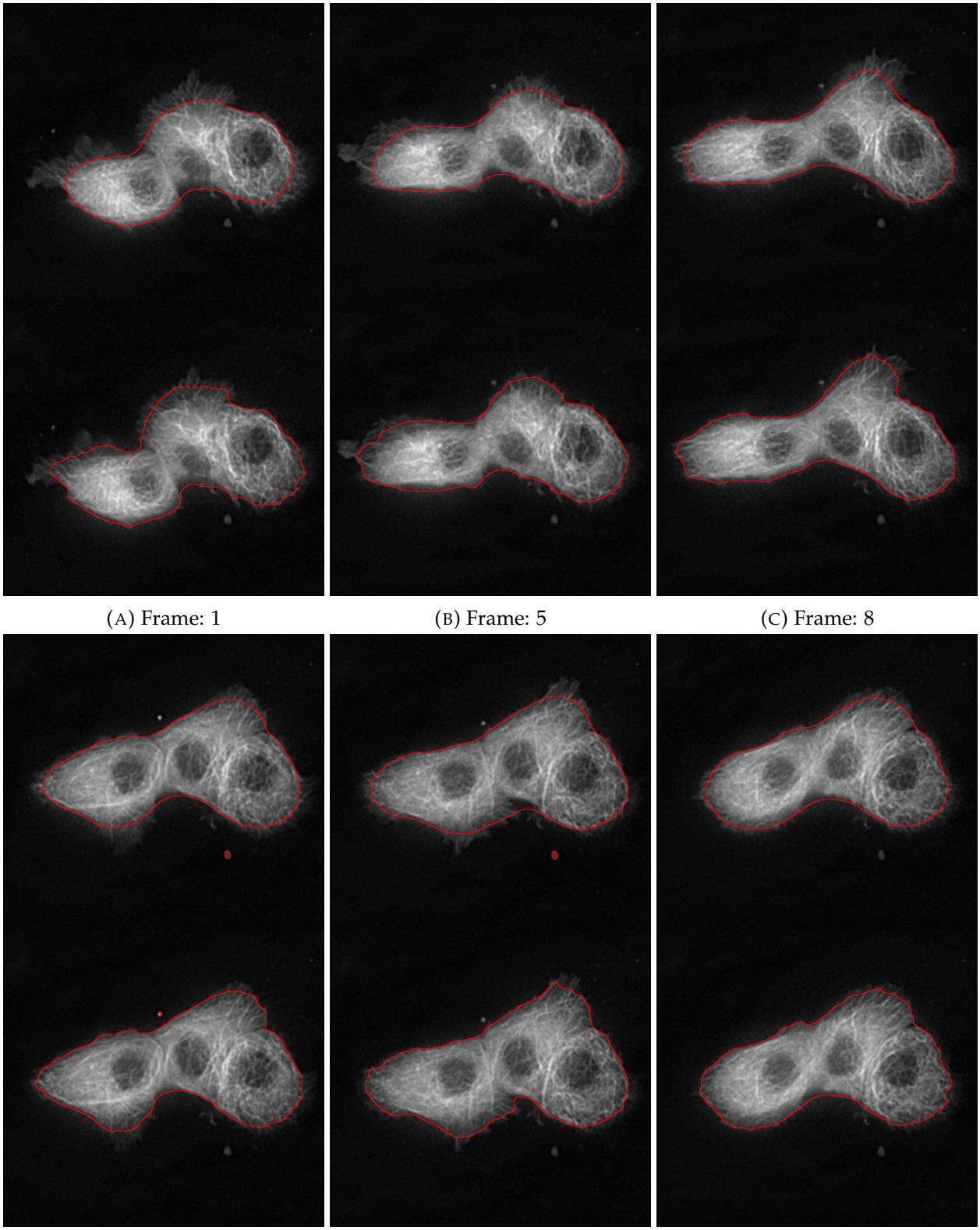


FIGURE 5.22: Comparing the segmentation results for the Epithilial Cells MFC 10A image set



(A) Frame: 1 (B) Frame: 5 (C) Frame: 8

(D) Frame: 17 (E) Frame: 20 (F) Frame: 30

FIGURE 5.23: Segmentation final results for the Epithilial Cells MFC 10A with one of the bigger improvements from manual to automatic tuning, from a participant with little experience in bioimage analysis. On top are the manual tuning results and on bottom the automatic ones.

Image Sequence 3: HeLa K cells expressing LMNA-GFP

For the third test we used an image set that shows HeLa K cells expressing LMNA-GFP [41], courtesy of Magdalena Gonciarz from the Mansfeld Lab of TU Dresden. The time sequence consisted of 31 frames and training was done on 3 key-frames (1,14,31).

In Figure 5.24 we can see that the average performance of the automatic tuning was much better than the manual. Also, there was no case in which the manual tuning was judged to have better results.

In Figure 5.25 we can see different frames from the image set with their segmentation outlines. The difference between the automatic and the manual tuning is clear, as the latter fails to segment a large number of cells. One other thing to observe is that there is much more movement in this case, we have cells that divide, increasing cell number. If we consider this an increase in the data variation, then it could explain the much better performance comparing to the other image sets. Another reason might be that this is in general a "tricky" image to segment as it has very big variation in brightness, and so finding good parameters with manual tuning can be difficult.

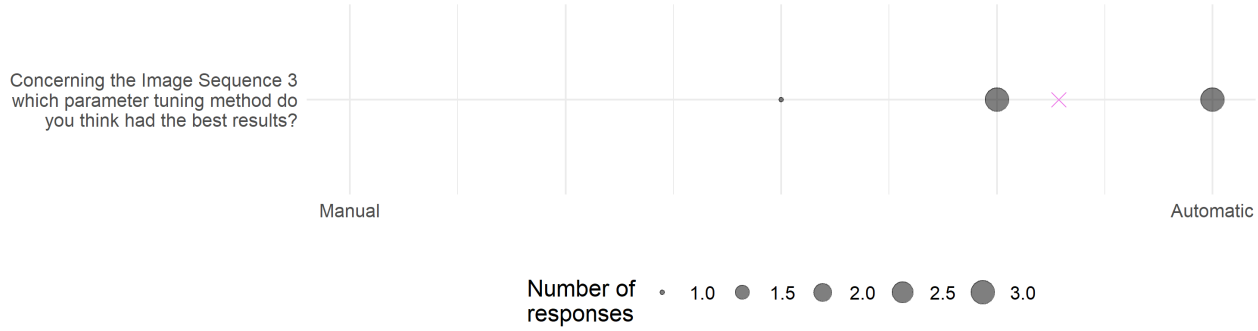
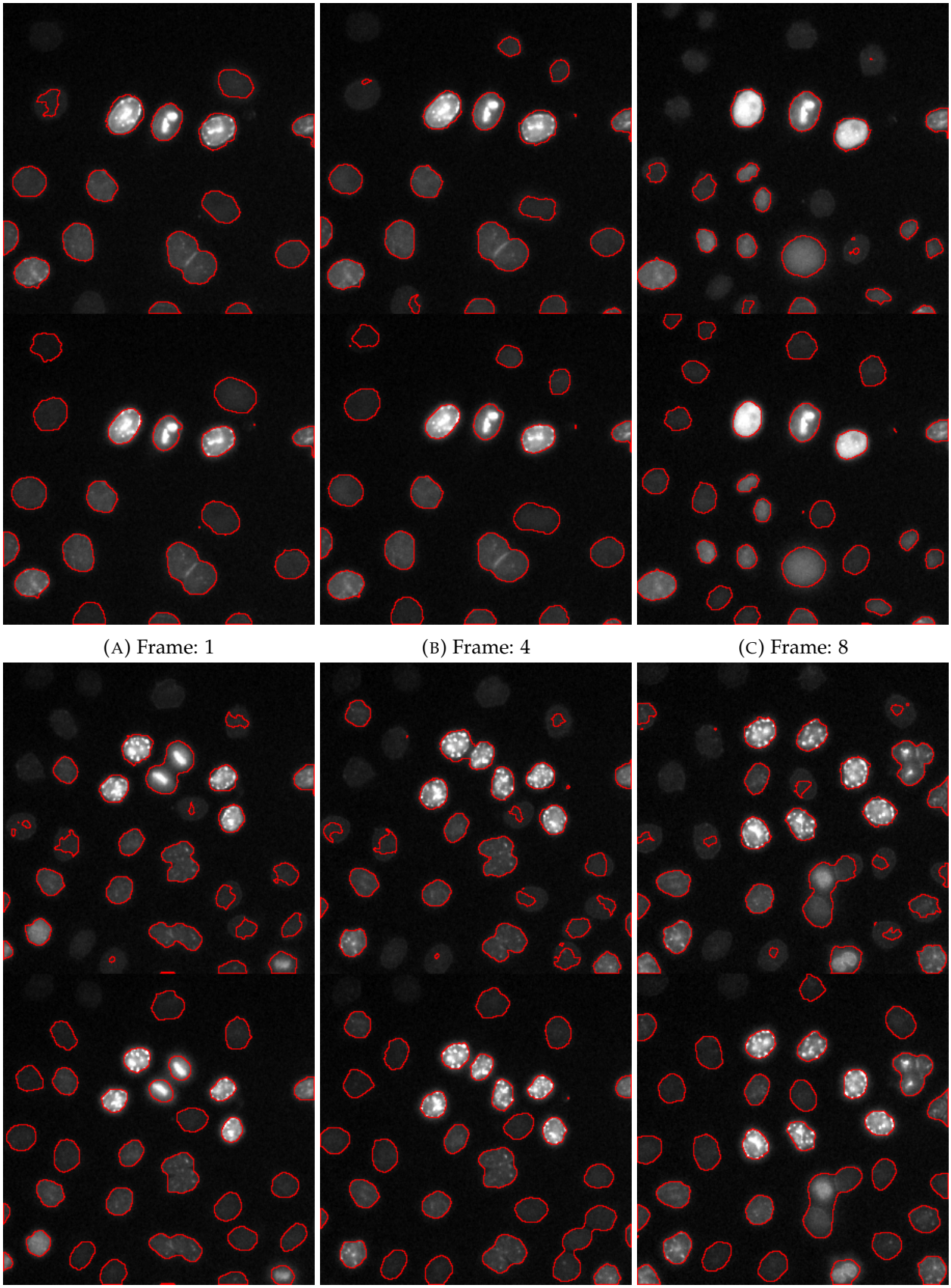


FIGURE 5.24: Comparing the segmentation results for the HeLa K cells expressing LMNA-GFP



(A) Frame: 1 (B) Frame: 4 (C) Frame: 8

(D) Frame: 13 (E) Frame: 17 (F) Frame: 31

FIGURE 5.25: Segmentation final results for the HeLa K cells expressing LMNA-GFP from the participant with the biggest improvement from manual to automatic tuning. On top are the manual tuning results and on bottom are the automatic ones.

5.3.5 Comparing Manual and Automatic Parameter Tuning

The next part of the Post-Questionnaire had some questions for comparing the two parameter tuning methods. In Figure 5.26 we can see these questions and their answers. In general, in all questions the automatic tuning method had an advantage. For example, on average the participants were more satisfied by the automatic tuning in terms of time and accuracy. Also, most participants said that the automatic tuning makes the segmentation procedure simpler and more accessible to non-expert users. Finally, there was one additional question in this section about the difficulty of manual tuning, and it was not surprising to see that most participants found it difficult.

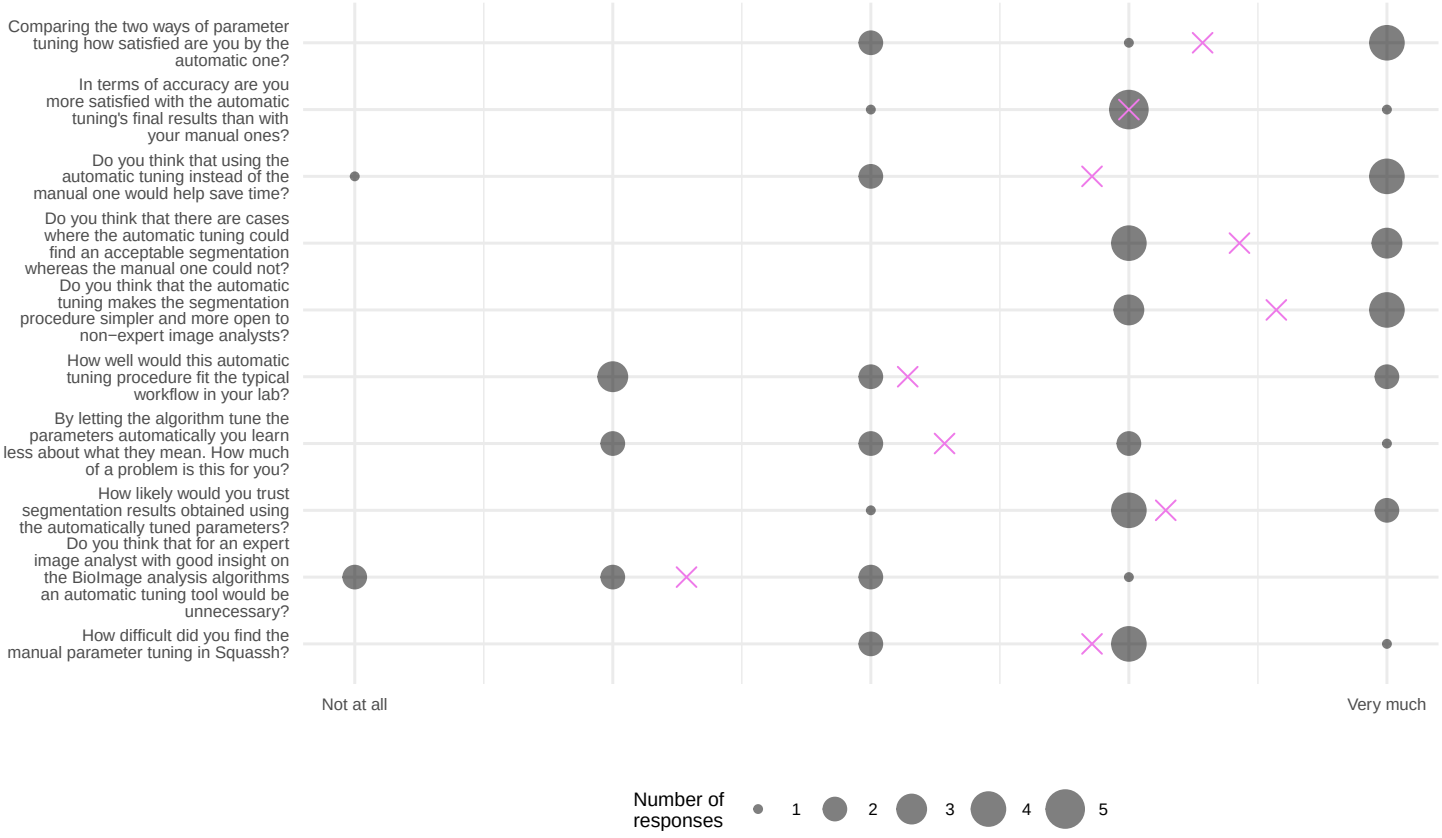


FIGURE 5.26: Comparing Manual and Automatic Parameter Tuning.

5.3.6 Before and after Questions

In the last part of the Post-Questionnaire we asked three questions concerning the Apeba plugin that were also asked in the Pre-Questionnaire, and one last question about using the plugin overall. The questions and their answers can be seen in Figure 5.27. In Figure 5.28 we can see the comparison of the before and after questions. Apparently, the experience from the usability test and the Apeba plugin made the users feel more comfortable to use an automatic tuning tool and more likely to trust its results.

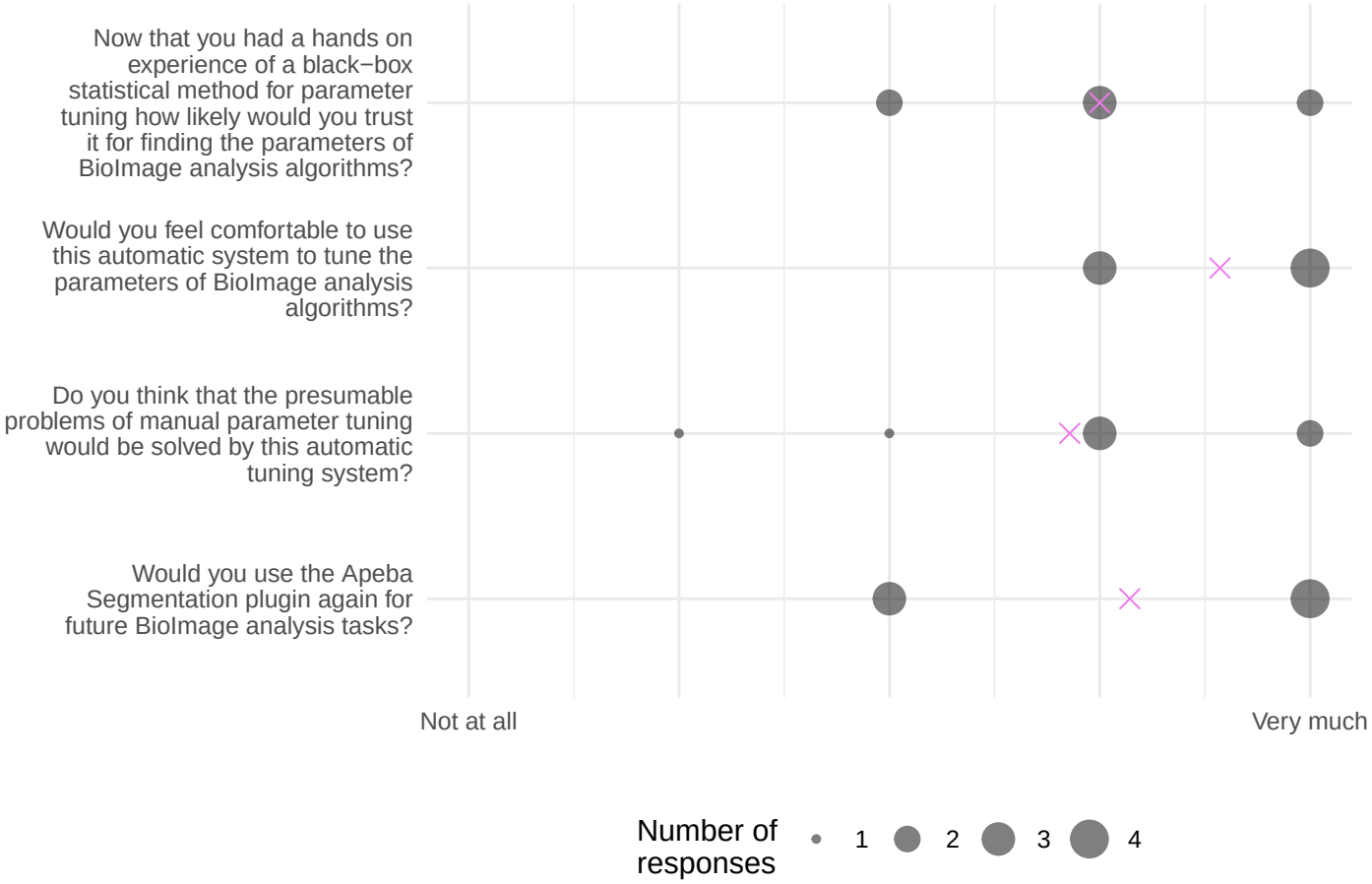


FIGURE 5.27: The last part of the Post-Questionnaire.

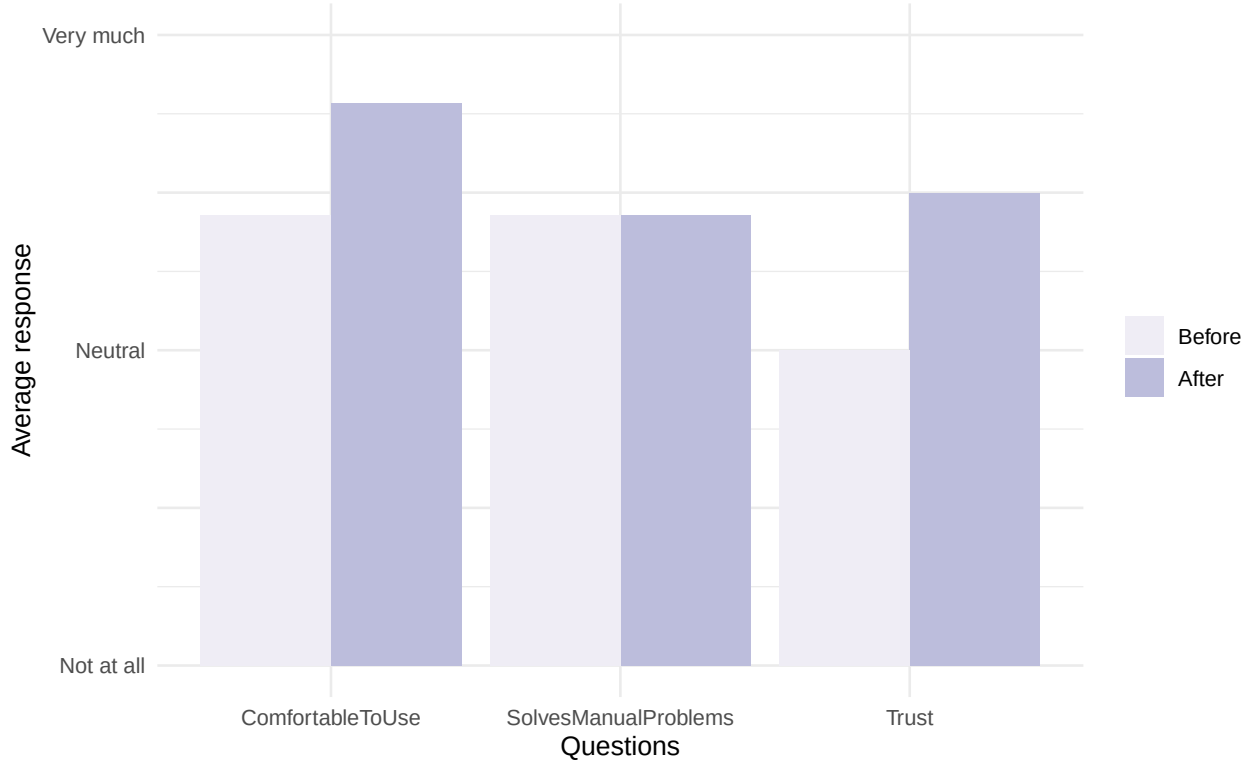


FIGURE 5.28: Before and after Questions. ComfortableToUse: "Would you feel comfortable to use this automatic system to tune the parameters of BioImage analysis algorithms?", SolvesManualProblems: "Do you think that the presumable problems of manual parameter tuning would be solved by this automatic tuning system?", Trust: "Now that you had a hands on experience of a black-box statistical method for parameter tuning how likely would you trust it for finding the parameters of BioImage analysis algorithms?"

6 Conclusion

6.1 Summary

The increasing volume and variation of data in modern biological experiments has created new needs for more automatic and robust software. As an effort to cope with these needs, we have presented an automatic parameter tuning software for bioimage analysis algorithms that focuses on giving robust solutions to the biological data variations based on user input. This software comes in the form of a Fiji plugin called Apeba, tuning the parameters of the Squassh segmentation algorithm. The plugin can be used for the tuning of any kind of image or sequence that is compatible with Squassh, while its training can be based on various types of information about the image or combinations of them (image count, object outlines, object markers etc). To our knowledge, this is the first time that the design centering problem is introduced in the bioimage analysis process for the tuning of a segmentation algorithm.

6.2 Main Results

We tested our automatic parameter tuning method in several examples and we made the following observations:

- Design centering has shown promising results in the automatic tuning of bioimage algorithms for image collections and different kinds of variations. From our tests we saw, that design centering outperformed the manual tuning 12 out of 15 times, while it outperformed the optimization tuning 11 out of 15 times. The improvement from the manual tuning was up to 45% and from CMAES up to 8% in the cases of foreground/background ground truth. For the cell counts, the maximum decrease of deviation from the ground truth was 72% for manual tuning and 41% for CMAES. Concerning the different variations, design centering has mainly shown better results for different fields of view, time series and z-stack, and mixed performance for different stains.
- The performance of the design centering tuning compared to the other two methods increases with more variation and complexity in the data. One example for this came from the "Drosophila Kc167 cells" image set where we saw that the increased number of cells in the samples resulted in bigger improvements for the design centering method. Furthermore, it was observed that decreasing the number of key-frames, which increases data variation, leads to better results for design centering compared to CMA-ES, indicating that indeed design centering finds more robust parameters.

- With regards to the manual tuning, we have seen that it is not considered an easy process by the users even for images of moderate difficulty. Additionally, there were many examples indicating that finding the best parameters for one frame of an image, can easily lead to bad results for the whole set.
- Approaching the parameter tuning task as an optimization problem has also shown good results. But we have seen that the more variation is added to the data, the more the accuracy falls. Another observation is that CMA-ES most of the times managed to find a better solution for the very key-frame than the design centering, as we would expect since it is an optimization algorithm. Therefore, the use of CMA-ES in data with little or no variation might be a better fit and a way to save time.
- Concerning the usability test, on average the participants preferred the results of the automatic tuning over the manual one. We observed that the plugin bridges the gap between the users with no experience and the experts, as their results using the plugin were equally good, contrary to the manual tuning. Furthermore, the before and after questions suggest that the experience with the Apeba plugin made the participants feel more comportable to use an automatic tuning tool and more likely to trust its results.
- The feedback about the user interface and experience with our plugin was very positive. It was nice to see that many participants commented the sub-oracle definition process as easy and "fun", while they found it also accessible to non-experienced users. However, we have to note that there are image cases where this process has limitations. For example, it is hard to give a drawing for very big or complex shapes, or give the object count of an image with hundreds of cells. Finally, giving outlines of objects with the use of a mouse can be inaccurate.
- Concerning the running time, we found out that the bottleneck of the process is the image analysis algorithm that is used. This means that the overall time can vary a lot depending on the algorithm that we choose to tune. Also, we should note the plugin has a lot of calls and data transfer between Java and MATLAB, which could be a reason for extra delay. For Squassh, the time for our tests varied from 40min to 10h. This is slow compared to other bioimage analysis methods, but it is important to keep in mind that the engagement of the user in the total process was usually not more than 10min.

  In the end, we can say that the use of the plugin is a trade-off between performance, manual tuning difficulty, and user experience in bioimage analysis on the one hand and time on the other hand.

6.3 Future Work

To conclude, we would like to highlight some thoughts and suggestions for future works. In this project, we implemented and tested the automatic tuning process only for the segmentation algorithm Squassh. Consequently, it would be interesting to extend the Apeba plugin to more bioimage analysis algorithms and run new tests for them. In this way, we could reach more firm conclusions for the design centering tuning method and also expand the cases that it can be handled. One idea would be to choose fast algorithms, e.g. pixel

classification methods, which would accelerate significantly the tuning process. By doing so, we could achieve a fully interactive software were the user can give information on an image, see right away the segmentation results updated, and repeat by improving the input for the training. Another suggestion for future work would be to test the design centering tuning method on larger image collections and benchmark over the size and variation of the data. Furthermore, keeping in mind that Squassh has only two parameters, we expect the benefit of automatic vs. manual tuning to grow for high-dimensional spaces. Also, in our tests we used fixed parameters for the Lp-Adaptation algorithm. Therefore, it is recommended to run more tests for various values of probability P (fixed or adapting), different population sizes, and different p-norms. To conclude, we consider that the results of our work in automatic parameter tuning are promising and believe that further development in this direction is worthwhile.

A User Interface

The following figures in this section are snapshots from the UI of the Apeba plugin for Fiji. Each UI page has the same structure containing the following areas:

- window description
- image area
- short description of the current task for the user
- 'Next' button
- 'Help' button

In figure A.1 we can see the wireframe of the UI and the position of these areas. The tools area has a different field for each page according to the current task. The image area has the original Fiji image dialog embedded, which enables to use the tools from the main Fiji UI (figure A.2). Normally, it is not needed to use the additional Fiji window because every tool is pre-selected for each page of the plugin.

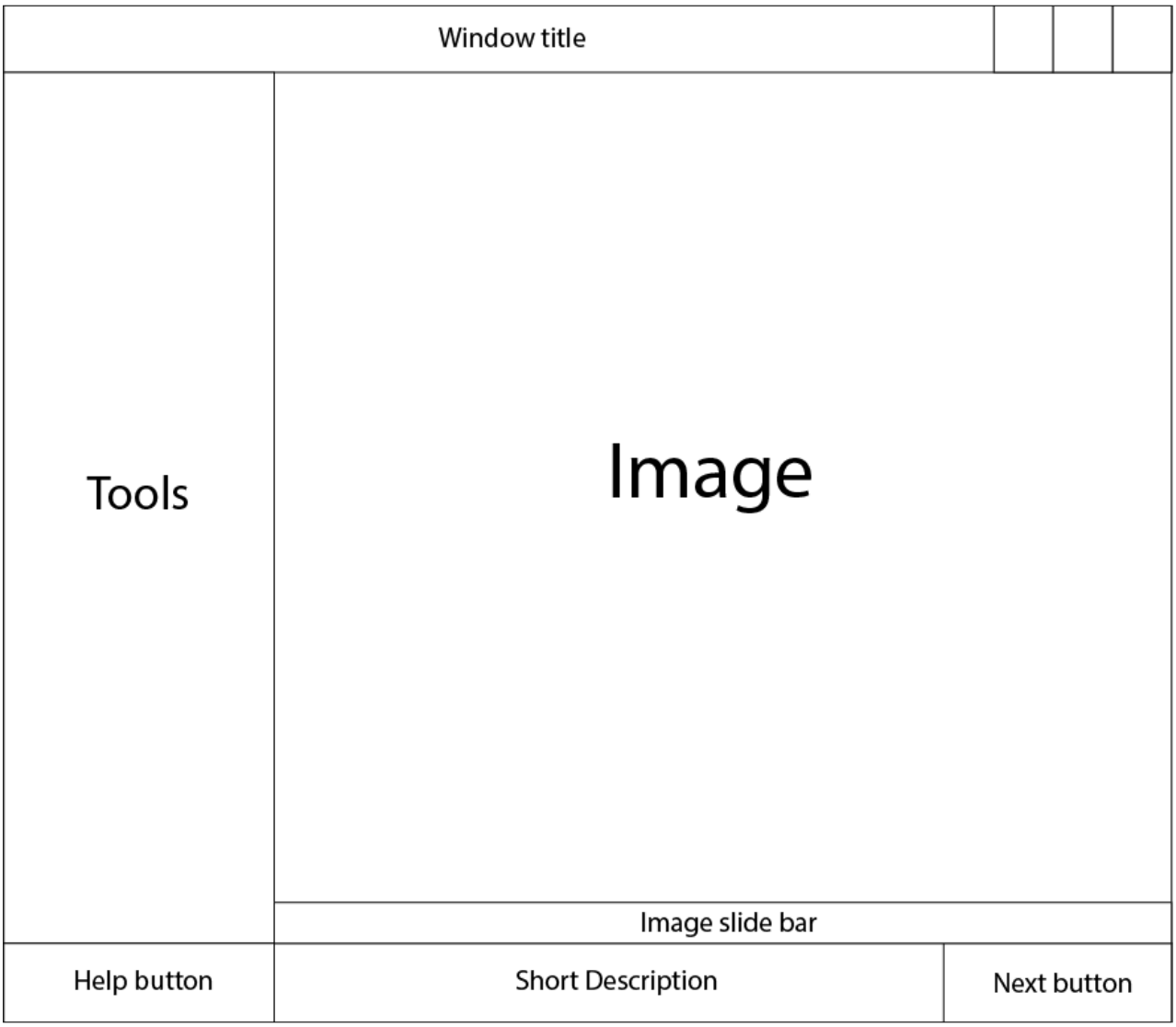


FIGURE A.1: Wireframe of the UI.

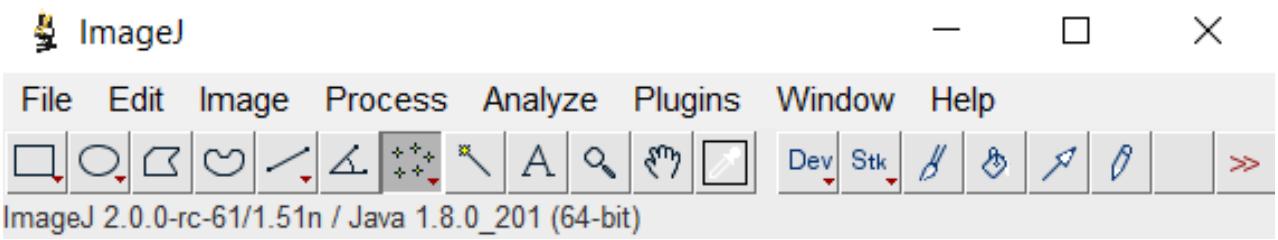


FIGURE A.2: Fiji main UI and toolbar.

FIGURE A.3: UI for choosing the key-frames of an image sequence. The user can navigate through the stack with the help of the slide bar below the image.

FIGURE A.4: The sub-oracles checklist UI.

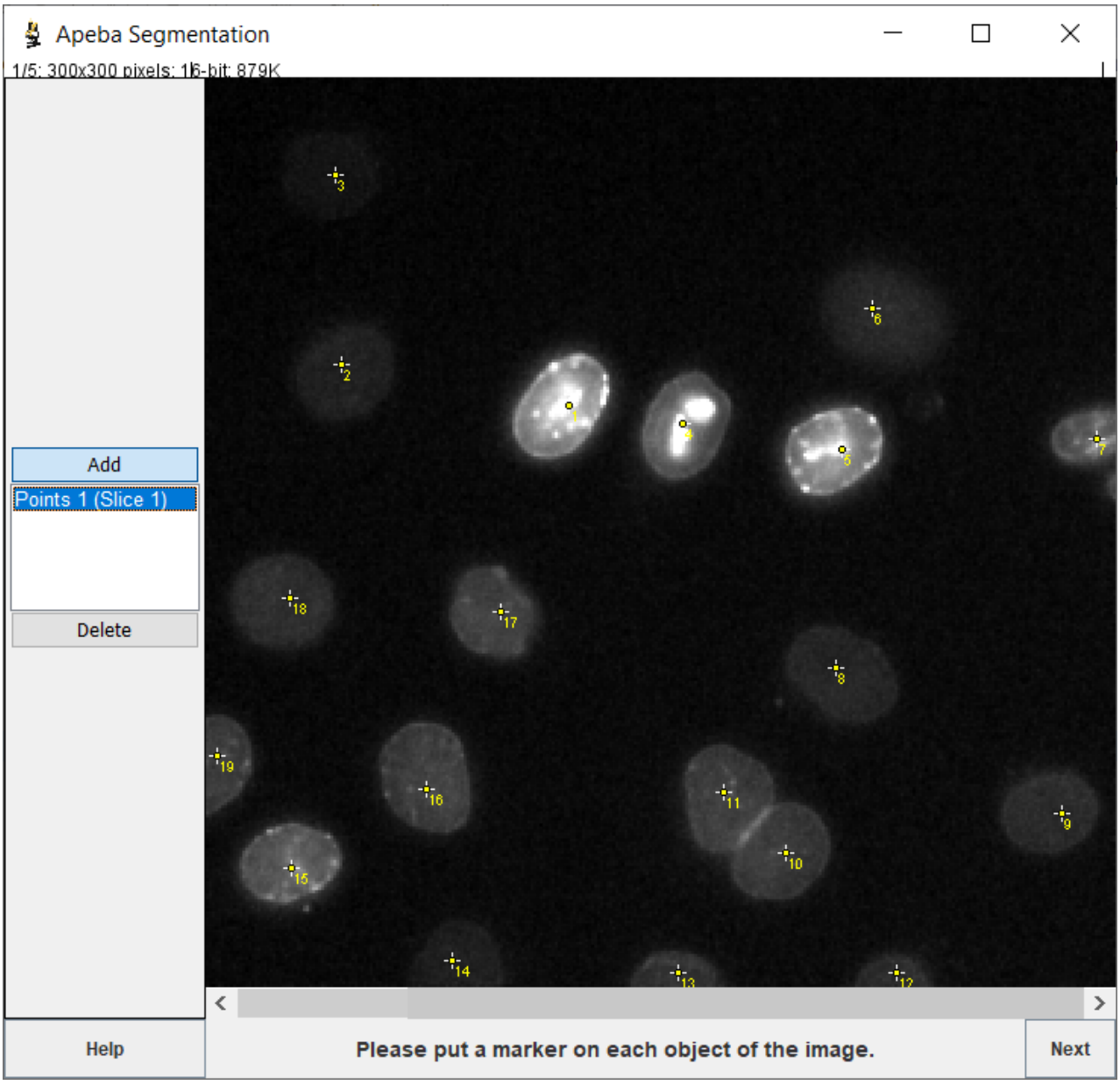


FIGURE A.5: The markers sub-oracle UI.

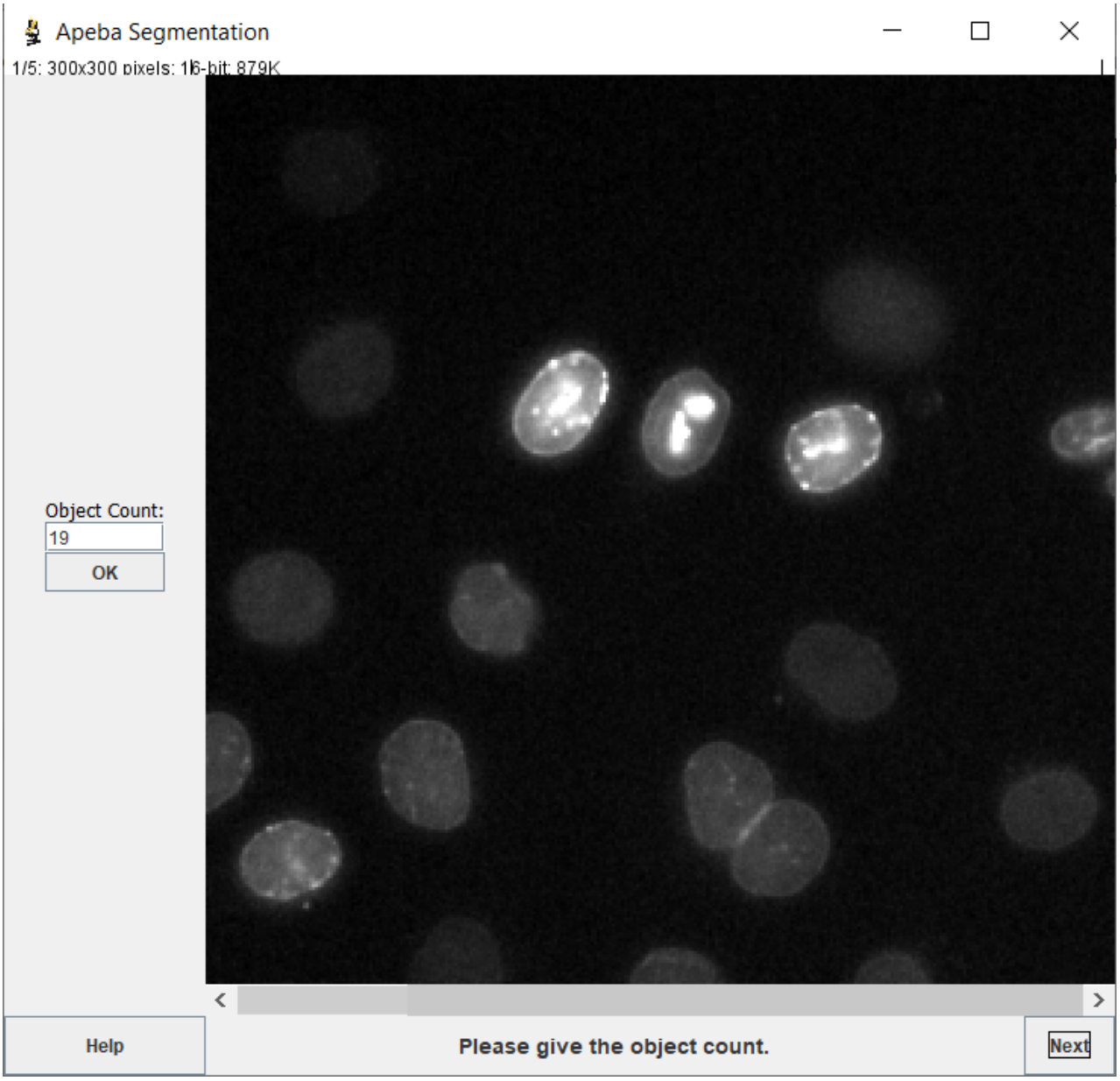


FIGURE A.6: The object number sub-oracle UI.

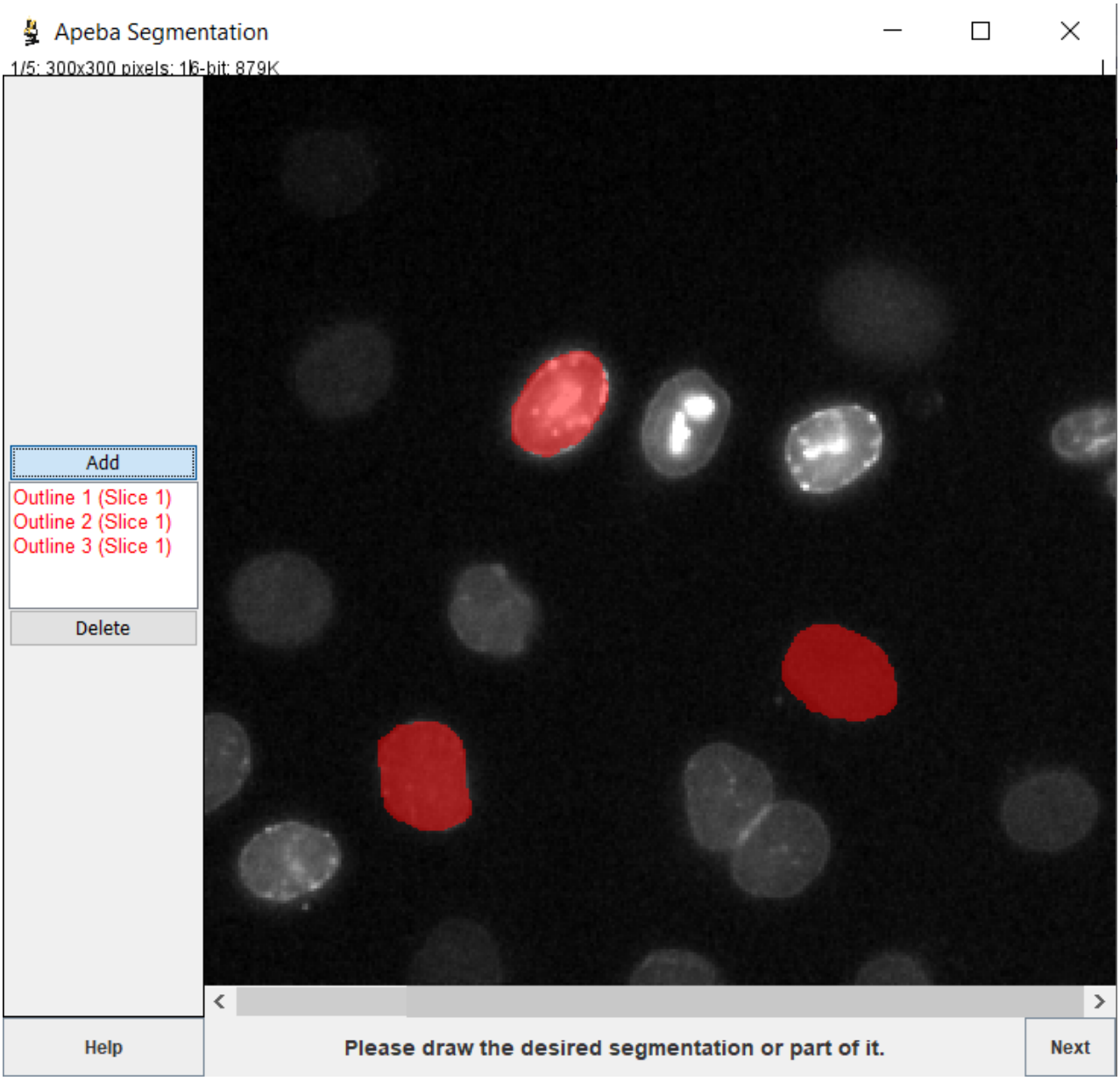


FIGURE A.7: The outline drawing sub-oracle.

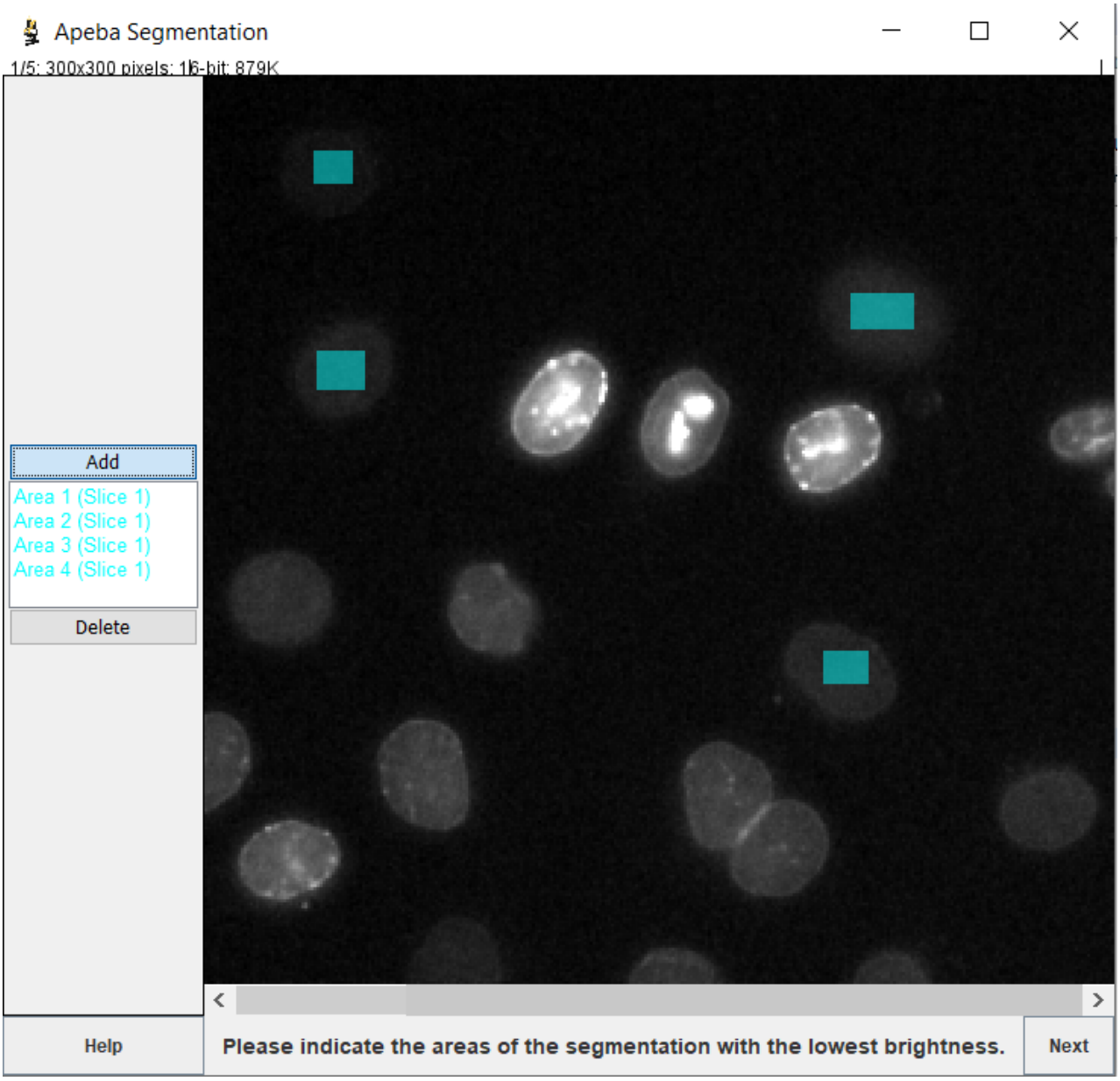


FIGURE A.8: The brightness sub-oracle.

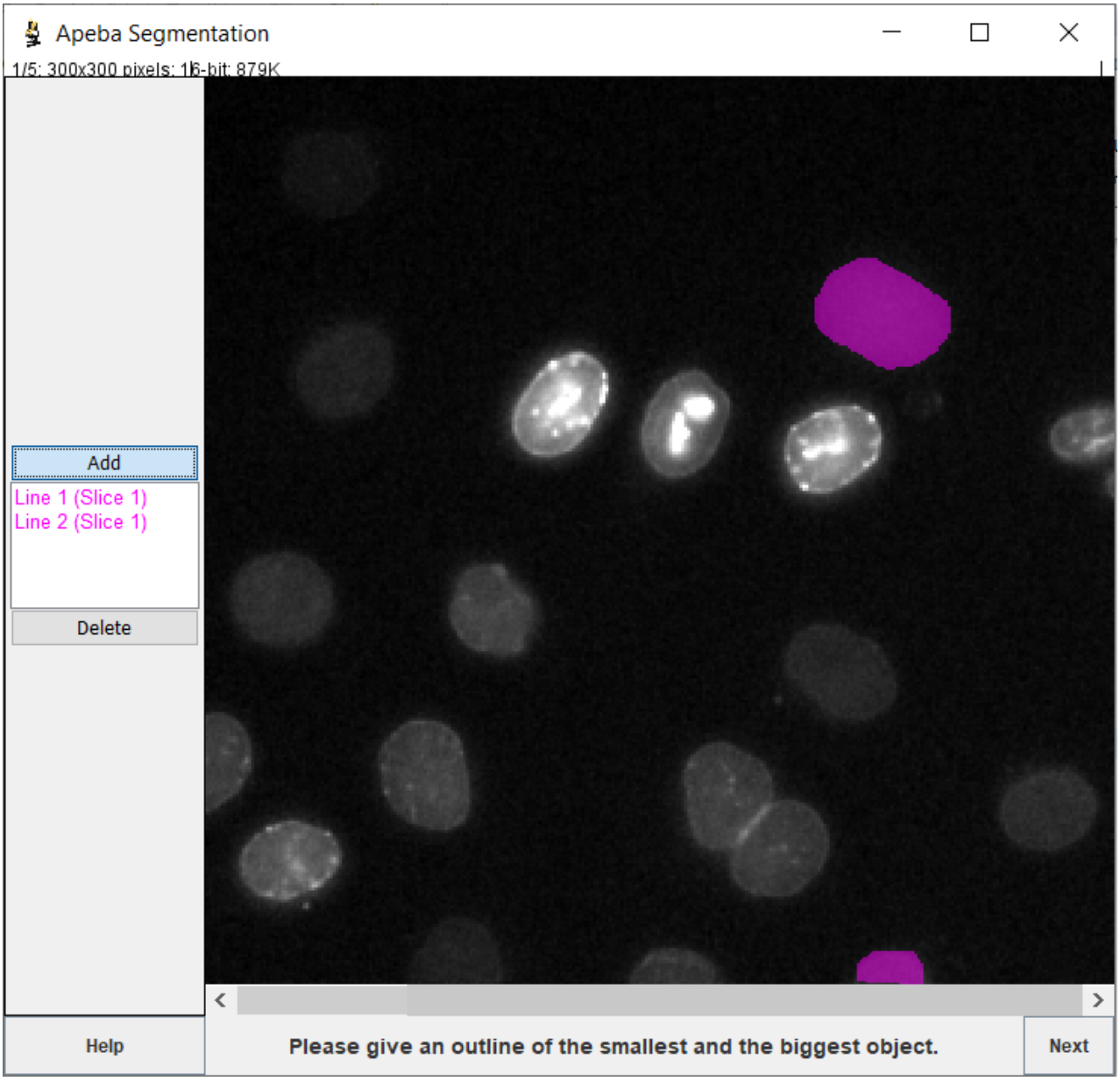


FIGURE A.9: The sizes sub-oracle

FIGURE A.10: The watershed sub-oracle.

B Usability Test Questions

Apeba Segmentation - Usability Test Pre-Questionnaire

This questionnaire was made for the purpose of a usability test for the Apeba Segmentation Fiji plugin (Automatic Parameter Estimation for BioIamge Analysis). The plugin is being developed at the MOSAIC Group (MPI-CBG) as part of the master thesis of Sotirios Piliouras with title "Automatic Parameter Learning for BioImage Analysis".

*Required

About you

1. **How experienced are you in BioImage Analysis?** *
   Mark only one oval.

	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	Very Experienced

2. **Have you used the Squassh Segmentation algorithm before?** *
   Mark only one oval.

   ◯ Yes
   ◯ No

3. **How experienced are you in manual parameter tuning for BioImage analysis algorithms?** *
   Mark only one oval.

	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	Very experienced

4. **Have you used an automatic parameter tuning tool for BioImage analysis algorithms previously?** *
   Mark only one oval.

   ◯ Yes
   ◯ No

Pre-Questionnaire

5. **How important is it for you to have absolute control of the BioImage analysis process?** *
Mark only one oval.

	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	Very important

6. **How often do you encounter problems with the manual parameter tuning of BioImage analysis algorithms?** *
Mark only one oval.

	1	2	3	4	5	
Never	◯	◯	◯	◯	◯	Very often

7. **Do you think that the presumable problems of manual parameter tuning would be solved by an automatic tuning system?** *
Mark only one oval.

	1	2	3	4	5	
Completely disagree	◯	◯	◯	◯	◯	Completely agree

8. **Would you feel comfortable to use an automatic system to tune the parameters of BioImage analysis algorithms?** *
Mark only one oval.

	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	Very comfortable

9. **How likely would you trust a black-box statistical method to tune the parameters of BioImage Analysis algorithms?** *
Mark only one oval.

	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	Very likely

Apeba Segmentation - Usability Test Questionnaire

This questionnaire was made for the purpose of a usability test for the Apeba Segmentation Fiji plugin (Automatic Parameter Estimation for BioIamge Analysis). The plugin is being developed at the MOSAIC Group (MPI-CBG) as part of the master thesis of Sotirios Piliouras with title "Automatic Parameter Learning for BioImage Analysis".

*Required

Usability

1. **How easy-to-use did you find the plugin?** *
   Mark only one oval.

	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	Very easy

2. **How often did you feel "stuck"?** *
   Mark only one oval.

	1	2	3	4	5	
Never	◯	◯	◯	◯	◯	All the time

3. **Did you find the 'Help' tips helpful?** *
   Mark only one oval.

	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	Very helpful

4. **How easy do you think it would it be for someone with no experience in BioImage Analysis to use the plugin?** *
   Mark only one oval.

	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	Very easy

5. **Do you think that by performing the same automatic tuning task again (given that you already had a first glance of the plugin), you could achieve better results?** *
   Mark only one oval.

	1	2	3	4	5	
Completely Disagree	◯	◯	◯	◯	◯	Completely Agree

Image Sequence 1: MC:BALB/c 3T3 Fibroblast

6. **Concerning the Image Sequence 1, which parameter tuning method do you think had the best results?** *
   Mark only one oval.

	1	2	3	4	5	
Manual	◯	◯	◯	◯	◯	Classic Automatic

7. **Concerning the Image Sequence 1, which parameter tuning method do you think had the best results?** *
   Mark only one oval.

	1	2	3	4	5	
Manual	◯	◯	◯	◯	◯	Stochastic Automatic

8. **Concerning the Image Sequence 1, which design centering method do you think had the best results?** *
   Mark only one oval.

	1	2	3	4	5	
Classic Automatic	◯	◯	◯	◯	◯	Stochastic Automatic

Image Sequence 2: Epithilial Cells MFC 10A

9. **Concerning the Image Sequence 2, which parameter tuning method do you think had the best results?** *
   Mark only one oval.

	1	2	3	4	5	
Manual	◯	◯	◯	◯	◯	Classic Automatic

10. **Concerning the Image Sequence 2, which parameter tuning method do you think had the best results?** *
    Mark only one oval.

	1	2	3	4	5	
Manual	◯	◯	◯	◯	◯	Stochastic Automatic

11. **Concerning the Image Sequence 2, which design centering method do you think had the best results?** *
Mark only one oval.

	1	2	3	4	5	
Classic Automatic	◯	◯	◯	◯	◯	Stochastic Automatic

Image Sequence 3: HeLa K cells expressing LMNA-GFP

12. **Concerning the Image Sequence 3, which parameter tuning method do you think had the best results?** *
Mark only one oval.

	1	2	3	4	5	
Manual	◯	◯	◯	◯	◯	Classic Automatic

13. **Concerning the Image Sequence 3, which parameter tuning method do you think had the best results?** *
Mark only one oval.

	1	2	3	4	5	
Manual	◯	◯	◯	◯	◯	Stochastic Automatic

14. **Concerning the Image Sequence 3, which design centering method do you think had the best results?** *
Mark only one oval.

	1	2	3	4	5	
Classic Automatic	◯	◯	◯	◯	◯	Stochastic Automatic

Comparing Automatic and Manual Parameter Tuning

15. **Comparing the two ways of parameter tuning, how satisfied are you by the automatic one?** *
Mark only one oval.

	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	Very satisfied

16. **In terms of accuracy, are you more satisfied with the automatic tuning's final results than with your manual ones?** *
Mark only one oval.

	1	2	3	4	5	
Completely disagree	◯	◯	◯	◯	◯	Completely agree

17. **Do you think that using the automatic tuning instead of the manual one would help save time?** *
Mark only one oval.

	1	2	3	4	5	
Completely disagree	◯	◯	◯	◯	◯	Completely agree

18. **Do you think that there are cases where the automatic tuning could find an acceptable segmentation whereas the manual one (due to time or other limitations) could not?** *
Mark only one oval.

	1	2	3	4	5	
Completely disagree	◯	◯	◯	◯	◯	Completely agree

19. **Do you think that the automatic tuning makes the segmentation procedure simpler and more open to non-expert image analysts?** *
Mark only one oval.

	1	2	3	4	5	
Completely disagree	◯	◯	◯	◯	◯	Completely agree

20. **How well would this automatic tuning procedure fit the typical workflow in your lab?** *
Mark only one oval.

	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	Very well

21. **By letting the algorithm tune the parameters automatically you learn less about what they mean. How much of a problem is this for you?** *
Mark only one oval.

	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	Very much

22. **How likely would you trust segmentation results obtained using the automatically tuned parameters?** *
Mark only one oval.

	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	Very likely

23. **Do you think that for an expert image analyst with good insight on the BioImage analysis algorithms an automatic tuning tool would be unnecessary?** *
Mark only one oval.

	1	2	3	4	5	
Completely disagree	◯	◯	◯	◯	◯	Completely agree

24. **How difficult did you find the manual parameter tuning in Squassh?** *
Mark only one oval.

	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	Very much

25. **Do you think that there are image sequences for which manual tuning would be more efficient in time or accuracy?** *
Mark only one oval.

- ◯ Yes
- ◯ No
- ◯ Maybe

26. **If yes, could you tell us more about these cases?**

27. **Do you have an idea of other image analysis algorithms that would be a good fit to the plugin? (for example algorithms that have difficult manual tuning, a lot of parameters, unpredictable behaviour etc)** *
Mark only one oval.

- ◯ Yes
- ◯ No

28. **If yes, please write what comes to your mind.**

29. **Now that you had a hands on experience of a black-box statistical method for parameter tuning, how likely would you trust it for finding the parameters of BioImage analysis algorithms?** *
Mark only one oval.

	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	Very likely

30. **Would you feel comfortable to use this automatic system to tune the parameters of BioImage analysis algorithms?** *
Mark only one oval.

	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	Very comfortable

31. **Do you think that the presumable problems of manual parameter tuning would be solved by this automatic tuning system?** *
Mark only one oval.

	1	2	3	4	5	
Completely disagree	◯	◯	◯	◯	◯	Completely agree

32. **Would you use the Apeba Segmentation plugin again for future BioImage analysis tasks?** *
Mark only one oval.

	1	2	3	4	5	
Definitely not	◯	◯	◯	◯	◯	Definitely

Bibliography

[1] Chris A Glasbey and Graham W Horgan. *Image analysis for the biological sciences*. Vol. 1. Wiley Chichester, 1995.

[2] Kevin W Eliceiri et al. "Biological imaging software tools". In: *Nature methods* 9.7 (2012), p. 697.

[3] David A Forsyth and Jean Ponce. *Computer vision: a modern approach*. Prentice Hall Professional Technical Reference, 2002.

[4] Steven W. Zucker. "Region growing: Childhood and adolescence". In: *Computer Graphics and Image Processing* 5.3 (1976), pp. 382–399. ISSN: 0146-664X. DOI: https://doi.org/10.1016/S0146-664X(76)80014-7. URL: http://www.sciencedirect.com/science/article/pii/S0146664X76800147.

[5] Erik Meijering et al. "Imagining the future of bioimage analysis". In: *Nature biotechnology* 34.12 (2016), p. 1250.

[6] Mehmet Sezgin and Bülent Sankur. "Survey over image thresholding techniques and quantitative performance evaluation". In: *Journal of Electronic imaging* 13.1 (2004), pp. 146–166.

[7] Dilpreet Kaur and Yadwinder Kaur. "Various image segmentation techniques: a review". In: *International Journal of Computer Science and Mobile Computing* 3.5 (2014), pp. 809–814.

[8] Robert M Haralick and Linda G Shapiro. "Image segmentation techniques". In: *Computer vision, graphics, and image processing* 29.1 (1985), pp. 100–132.

[9] Christian Held et al. "Approaches to automatic parameter fitting in a microscopy image segmentation pipeline: An exploratory parameter space analysis". In: *Journal of pathology informatics* 4.Suppl (2013).

[10] A Johannes Pretorius et al. "Visualization of parameter space for image analysis". In: *IEEE Transactions on Visualization and Computer Graphics* 17.12 (2011), pp. 2402–2411.

[11] Christoph Sommer et al. "Ilastik: Interactive learning and segmentation toolkit". In: *2011 IEEE international symposium on biomedical imaging: From nano to macro*. IEEE. 2011, pp. 230–233.

[12] Anne E Carpenter et al. "CellProfiler: image analysis software for identifying and quantifying cell phenotypes". In: *Genome biology* 7.10 (2006), R100.

[13] Ignacio Arganda-Carreras et al. "Trainable Weka Segmentation: a machine learning tool for microscopy pixel classification". In: *Bioinformatics* 33.15 (2017), pp. 2424–2426.

[14] *Interactive Watershed*. https://imagej.net/Interactive_Watershed.

[15] Josefine Asmus, Christian L Müller, and Ivo F Sbalzarini. "Lp-Adaptation: Simultaneous Design Centering and Robustness Estimation of Electronic and Biological Systems". In: *Scientific reports* 7.1 (2017), p. 6660.

[16] Sachin S Sapatnekar, Pravin M Vaidya, and Sung-Mo Kang. "Convexity-based algorithms for design centering". In: *IEEE transactions on computer-aided design of integrated circuits and systems* 13.12 (1994), pp. 1536–1549.

[17] Josefine Asmus. "An Efficient Randomized Approximation Algorithm for Volume Estimation and Design Centering". In: (2017).

[18] G Kjellstrom and Lars Taxen. "Stochastic optimization in system design". In: *IEEE Transactions on Circuits and Systems* 28.7 (1981), pp. 702–715.

[19] Hany L Abdel-Malek and A-KSO Hassan. "The ellipsoidal technique for design centering and region approximation". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 10.8 (1991), pp. 1006–1014.

[20] S Director and G Hachtel. "The simplicial approximation approach to design centering". In: *IEEE Transactions on Circuits and Systems* 24.7 (1977), pp. 363–372.

[21] Chenjie Gu and Jaijeet Roychowdhury. "Yield estimation by computing probabilistic hypervolumes". In: *Extreme Statistics in Nanoscale Memory Design*. Springer, 2010, pp. 137–177.

[22] Rainer Storn and Kenneth Price. "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces". In: *Journal of global optimization* 11.4 (1997), pp. 341–359.

[23] Abbas Seifi, K Ponnambalam, and Jiri Vlach. "A unified approach to statistical design centering of integrated circuits with correlated parameters". In: *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* 46.1 (1999), pp. 190–196.

[24] RS Soin and R Spence. "Statistical exploration approach to design centring". In: *IEE Proceedings G (Electronic Circuits and Systems)*. Vol. 127. 6. IET. 1980, pp. 260–269.

[25] Sheng Liu, Jian Zhang, and Binhai Zhu. "Volume computation using a direct Monte Carlo method". In: *International Computing and Combinatorics Conference*. Springer. 2007, pp. 198–209.

[26] Robert Schwencker et al. "The generalized boundary curve-A common method for automatic nominal design and design centering of analog circuits". In: *Proceedings Design, Automation and Test in Europe Conference and Exhibition 2000 (Cat. No. PR00537)*. IEEE. 2000, pp. 42–47.

[27] Chris P Barnes et al. "Bayesian design of synthetic biological systems". In: *Proceedings of the National Academy of Sciences* 108.37 (2011), pp. 15190–15195.

[28] Johannes Schindelin et al. "Fiji: an open-source platform for biological-image analysis". In: *Nature methods* 9.7 (2012), p. 676.

[29] Caroline A Schneider, Wayne S Rasband, and Kevin W Eliceiri. "NIH Image to ImageJ: 25 years of image analysis". In: *Nature methods* 9.7 (2012), p. 671.

[30] ImageJ. *Writting Plugins*. `https://imagej.net/Writing_plugins`.

[31] Nikolaus Hansen and Andreas Ostermeier. "Completely derandomized self-adaptation in evolution strategies". In: *Evolutionary computation* 9.2 (2001), pp. 159–195.

[32] Nikolaus Hansen, Sibylle D Müller, and Petros Koumoutsakos. "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)". In: *Evolutionary computation* 11.1 (2003), pp. 1–18.

[33] Aurélien Rizk et al. "Segmentation and quantification of subcellular structures in fluorescence microscopy images using Squassh". In: *Nature protocols* 9.3 (2014), p. 586.

[34] Stanley R Sternberg. "Biomedical image processing". In: *Computer* 1 (1983), pp. 22–34.

[35] *Broad Bioimage Benchmark Collection*. `https://data.broadinstitute.org/bbbc/`.

[36] *Cell Image Library*. `http://www.cellimagelibrary.org/home`.

[37] Vebjorn Ljosa, Katherine L Sokolnicki, and Anne E Carpenter. "Annotated high-throughput microscopy image sets for validation." In: *Nature methods* 9.7 (2012), pp. 637–637.

[38] Vladimir Ulman et al. "An objective comparison of cell-tracking algorithms". In: *Nature methods* 14.12 (2017), p. 1141.

[39] Maria Nemethova. "Carassius auratus auratus, fibroblast." In: *Olympus BioScapes Digital Imaging Competition® (2012) CIL:42607* (2009). URL: https://doi.org/doi:10.7295/W9CIL42607.

[40] David Svoboda, Michal Kozubek, and Stanislav Stejskal. "Generation of digital phantoms of cell nuclei and simulation of image formation in 3D image cytometry". In: *Cytometry Part A: The Journal of the International Society for Advancement of Cytometry* 75.6 (2009), pp. 494–509.

[41] Magdalena Gonciarz. "HeLa K cells expressing LMNA-GFP". In: *Mansfeld Lab TU Dresden* ().