Thèse préparée à l'Institut Pasteur de Paris et à Télécom Paris
delivrée par Sorbonne Université

École Doctorale Informatique, Télécommunications, Électronique de Paris n°130
Unité d'Analyse d'Images Biologiques - UMR 3691

Approche Bayésienne Intégrant le Flux Optique pour le Suivi d'Objets Multiples en Imagerie Biologique.

Application au Suivi de Neurones dans l'Hydre.

Integrating Optical Flow within a Bayesian Framework for Multiple Object Tracking in Biological Imaging.

Application to Neuron Tracking in Hydra.

Par Raphaël RÈME

Thèse de doctorat d'Informatique

Dirigée par Thibault LAGACHE et co-encadrée par Jean-Christophe OLIVO-MARIN, Elsa ANGELINI et Alasdair NEWSON

Présentée et soutenue publiquement le 1er décembre 2025

Devant un jury composé de

Michal KOZUBEK	Professor, Masaryk University	Rapporteur
Ihor SMAL	Assistant Professor, Utrecht University	Rapporteur
Anna KRESHUK	Group Leader, EMBL Heidelberg	Examinatrice
Auguste GENOVESIO	Directeur de Recherche, ENS Paris	Président du Jury
Perrine PAUL-GILLOTEAUX	Ingénieure de Recherche, Université de Nantes	Examinatrice
Jean-Yves TINEVEZ	Ingérieur de Recherche, Institut Pasteur	Invité
Alasdair NEWSON	Maître de Conférences, Sorbonne Université	Co-encadrant
Elsa ANGELINI	Professeure, Télécom Paris	Co-encadrante
Jean-Christophe OLIVO-MARIN	Professeur, Institut Pasteur	Co-encadrant
Thibault LAGACHE	Chargé de Recherche, Institut Pasteur	Directeur de thèse

Remerciements

Cela fait maintenant 3 ans et quelques (8) mois que je travaille sur mon sujet de thèse. Durant ce long mais passionnant parcours, j'ai été soutenu par de nombreuses personnes, dont mes proches, sans qui je ne serais sans doute pas venu au bout de cette épreuve. J'aimerais donc prendre un moment pour vous remercier, vous qui m'avez accompagné pendant ces années, celles d'avant et, je l'espère, celles d'après.

Tout d'abord, je tenais à remercier les membres de mon jury pour le temps qu'ils ont accordé à ce projet. Merci pour vos critiques constructives et bienveillantes qui ont mené à une discussion scientifique enrichissante. En particulier, je voudrais remercier les deux rapporteurs, Ihor et Michal, qui ont eu la longue (et je l'espère, pas trop difficile) tâche de lire attentivement ce manuscrit. Merci pour vos précieux retours.

Je voudrais ensuite remercier mes encadrants qui ont rendu ce sujet possible. Avoir 4 encadrants, c'était sûrement une folie, entre surplus de réunions, problèmes de communication et avis divergents. Mais j'ai aussi toujours eu quelqu'un à qui parler et pour m'aider sur les divers sujets qu'on abordait. J'ai pu apprendre des choses différentes auprès de chacun d'entre vous et votre aide a permis de faire de cette thèse ce qu'elle est aujourd'hui. Alasdair, merci d'avoir rejoint ce projet après quelques bières et une présentation bancale, alors que nous ne savions pas encore où nous allions. On n'a pas vraiment fait le DL qu'on pensait faire initialement, mais tu es resté et m'as soutenu pendant tout ce projet. Grâce à toi, je peux enfin écrire mes maths correctement (enfin je l'espère). Elsa, merci pour ton soutien et nos discussions tout au long de cette thèse. Tu m'as donné l'occasion d'encadrer des cours à Télécom, de participer à des conférences et de faire de super rencontres. Merci pour tes retours (généralement le week-end à 23:30 !) et tes idées sur tous les projets qu'on a menés à bien grâce à toi. Jean-Christophe, merci pour ton accueil dans le labo à Pasteur : je me suis toujours senti à ma place dans cette équipe que tu as créée. Merci aussi pour toutes nos discussions « tracking », sujet qui te tient à coeur autant qu'à moi. Tu m'as aussi permis de progresser pour trouver les bons axes et les bonnes formulations pour présenter mes travaux, que ce soit dans nos papiers ou mes talks (et je partais de loin). Enfin, merci Thibault d'avoir dirigé ce projet avec bonne humeur et merci pour ces blagues (que je ne pourrais pas retranscrire ici) qui te définissent. Tu as toujours suivi le projet de près sans jamais me mettre la pression et j'ai toujours pu compter sur ton aide pour réécrire mes premiers jets (qui avaient bien besoin de tes retouches). Tout ce travail n'existerait pas sans toi, merci d'avoir suivi et soutenu toutes mes idées parfois farfelues.

J'aimerais ensuite remercier toute mon équipe à Pasteur, au BIA. Merci à Nancy et à nos discussions en dîner de labo, à Vannary pour ton assistance et tes cookies, à Élisabeth, Giacoma et Maria pour les nombreux pots et délires qu'on a partagés (avec sans aucun doute ce même besoin de noyer son chagrin), et enfin à Pascal pour tes histoires toujours plus folles. Un très très grand merci à Marie-Anne et Olivia pour votre gentillesse et votre soutien sans faille durant tout le long de cette thèse. Grâce à vous mon moral n'a jamais laché, votre aide m'a été inestimable.

Dans ce laboratoire, j'ai aussi eu la chance de partager mes galères de thésard avec une formidable équipe de doctorants. Entre PMU, raclettes, skis, randos, escalades, et un petit peu de sciences, je vous remercie tous de m'avoir accompagné. Merci à Alexandre et Diana, vestiges d'un passé lointain mais pourtant toujours dans les bons coups. Merci pour nos discussions vélo après piscine de pinard, Alex. Diana, merci pour nos soirées colombiennes endiablées à Carthagène ou à Paris ! Ensuite, merci à Samuel, premier souffre-douleur de Thibault, de m'avoir montré qu'on pouvait sérieusement le vouvoyer pendant plusieurs années. Merci pour toutes nos séances de bad où tu m'as toujours laissé gagner. Merci pour les courses, randos et bières, ainsi que pour toutes les bêtises que tu as pu dire et les fous rires. Yekta, de pair avec Samuel (vous vous êtes vraiment bien trouvés), merci pour ta gentillesse inébranlable et nos séances de travail productives pour maîtriser l'ouverture italienne. Toujours plus de fous rires sur une de tes expressions d'un autre temps ! Pour te paraphraser, merci de m'avoir permis de vivre ces moments privilégiés en ta companie. À l'intersection entre Télécom et Pasteur (et le badminton bien sûr), un grand merci à Erwan et Antoine de m'avoir ouvert la voie au long de ces 3 années. Sans vous, elles auraient été bien plus ternes. Erwan, merci pour tous les moments passés ensemble depuis le début de cette thèse. Notamment nos « vacances » en Colombie ou en Grèce, nos randos au Queyras, et toutes ces soirées folles. Hâte de la prochaine soirée déguisée avec toi. J'ai toujours pu compter sur ton aide quand j'en avais besoin, merci du fond du coeur. Antoine, après 8 ans (tu dates un peu), que te dire ? Trois années supplémentaires de souvenirs dingues à tes côtés entre nez rouge, Ranrok à ski, grande-tante et 50 kg de légumes. Merci pour tout. Merci ensuite à Benjamin et Mounib pour votre pugnacité exemplaire et nos discussions de geeks que j'affectionne tant autour d'une bière ou d'un café. Mounib, bon courage pour ta dernière ligne droite et Benjamin, bravo pour la tienne. Merci à Thomas & Thomas. Merci Thomas pour toutes nos parties de Uno, et nos bières partagées après le boulot. Merci à Thomas et tes rhums frelatés, toujours debout à la fin de soirée (enfin, quoique...). Merci notamment pour cette super retraite départementale. Enfin, merci à la nouvelle équipe de doctorants qui a redonné un sens aux pauses café et a remonté mon moral de fin de thèse. En particulier, merci à Sarah pour ta gentillesse et ta folie cachée. Hâte de venir à ta prochaine pièce de théâtre. Merci à Joséphine pour ces moments et discussions autour d'un café ou en TP à Télécom. Quel bel exemple tu nous donnes de maturité et d'équilibre, merci. Hate de la prochaine session escalade. Tristan, merci pour nos sessions de ski passées et celles à venir, et merci d'être toujours présent pour les verres à l'improviste, et de manière générale merci d'être là pour n'importe quelles activités random qu'on pourrait avoir. Bon courage pour la reprise du tracking, compte sur moi pour te donner un coup de main ! Merci à Julie pour son grand soutien moral en cette fin de thèse et tous ses projets artistiques. Merci d'être cette force motrice pour les pots et bières ! Et merci de m'accompagner en grinchant tous les mercredis au badminton. Enfin, merci à Valentina pour ces trois conférences passées ensemble en Colombie, Grèce et Houston, j'en garde d'excellent souvenirs comme tu l'imagines. J'espère qu'on aura l'occasion de tous se retrouver à Londres cette année.

Merci ensuite à l'équipe IMAGES à Télécom qui m'a accueilli pendant ces trois années à côté de Pasteur. En particulier, j'aimerais remercier Yann, Pietro, Saïd, Christophe et Loïc pour toutes nos discussions et pour les TPs que vous m'avez permis de faire. Merci à toute l'équipe de doctorants de Télécom : Inès, Zoé, Nicolas, Pierrick, Rebecca, Emma, Erwan, Antoine, Marie, Chloé, Gwilherm, Édouard, Rassim, Robin, Ali & Ali, Christiano, Thomas, Victor, Sélim, Edgar, Stella, Quentin, Indira. Si Saclay n'était pas aussi loin, je serais venu plus souvent car c'était toujours un plaisir de passer du temps avec vous. Merci pour ces soirées Divan du Monde de folie et petites bières à l'ancienne au bar de Tetech.

A côté de ma thèse, j'ai aussi eu la chance de pouvoir faire du badminton et du volley

avec Pasteur. Je voulais donc chaleureusement remercier toute l'équipe du bad du mercredi et vendredi et toute l'équipe du volley du lundi. Vous m'avez véritablement permis de tenir le coup, merci beaucoup.

Maintenant j'aimerais remercier mes amis. Merci à mes amis d'enfance d'avoir toujours été là aussi loin que je m'en souvienne et de continuer à partager ma vie aujourd'hui. Merci à Léo, Lukas, Raph et QQ d'avoir fait le déplacement pour la soutenance. Merci à mes amis de prépa et d'école qui ont dû supporter mes nombreux excès, mais, souvent aussi, qui les ont partagés avec un peu trop d'enthousiasme. Merci Ugo, Cess, Gabos, Marie, Robin, GJ, Gab, Ariane, Manu, Mathu, Marine d'être venu à la soutenance. Une grosse pensée à Yasmine qui j'en suis sur serait venu avec plaisir. Merci pour toutes ces vacances, ces bouffes, et ces randos, pour la fête et pour tous ces moments qu'on a eus ensemble. Merci d'être restés à mes côtés, notamment, quand ça n'allait pas, et que je n'avais plus la tête à ça.

Un grand merci à toute ma famille. Grand-Mère, merci pour tous tes sourires et tous ces souvenirs à Seyssin que je chéris. J'aurais tant aimé que Grand-Père soit encore là avec toi aujourd'hui pour voir ça. Merci à Bon-Papa et Bonne-Maman pour leur sens de la famille qu'ils m'auront transmis et à Bon-Papa pour sa passion des problèmes, avec ses sudokus au coin du feu. Merci à mes oncles et tantes, mes cousins pour toutes ces vacances en famille en Corse, à Sauldre et ailleurs. Merci à ma belle-famille de m'avoir si bien accueilli, je suis chanceux de vous avoir désormais à mes côtés. Enfin, Pap', Mam', merci de m'avoir tout appris et toujours aidé à travers les années, quels que soient mes problèmes. Je sais que je peux toujours compter sur vous. Désolé pour ces 3 années difficiles où je n'ai pas toujours pu être très présent. Enfin, merci à mes deux sistas, LauLau, Juju. Moi qui étais un petit geek un peu perdu, merci de m'avoir fait ouvrir les yeux sur le monde et de continuer à le faire.

Pour finir, je voudrais te remercier Soso. Tu as été aux premières loges de mes galères depuis les 5 dernières années et malgré ça tu es restée et tu as supporté mes plaintes et mes pleurs. Et que de bons souvenirs avec toi pendant cette thèse, avec tous nos voyages à vélo, randos et découvertes du monde ensemble. J'ai vraiment hâte de vivre les prochaines années à tes côtés et de voir où elles nous mèneront. Je t'aime.

Cette thèse, c'est avant tout le fruit de mon environnement de travail et social, et de mon éducation. Merci à tous d'y avoir participé.

Abstract

Abstract: Understanding neuronal activity in behaving animals requires accurate tracking of large neuronal populations over time. In small model organisms such as *Hydra vulgaris*, *Caenorhabditis elegans* or zebrafish, calcium imaging enables the monitoring of hundreds of neurons simultaneously. However, reliable tracking, the first step of the automatic image analysis pipeline, remains a major challenge due to several factors: strong body deformations, dense neuronal populations, imaging noise, and the intermittent detectability of neurons when using calcium indicators (GCaMP). Current algorithms often fail in these conditions, producing fragmented trajectories or incorrect associations that compromise biological interpretation.

The goal of this thesis is to develop robust computational tools for multiple object tracking (MOT) adapted to these biological challenges. Our contributions are structured along three complementary axes:

(1) Simulation framework (SINETRA): To overcome the lack of annotated experimental datasets, we developed SINETRA, a versatile simulator of neurons embedded in deformable tissues. It generates realistic 2D/3D fluorescence videos with ground-truth annotations, reproducing motion patterns and noise similar to those observed in behaving animals. This synthetic benchmark enables objective evaluation and comparison of neuron tracking algorithms.

(2) Tracking with optical flow enhanced motion models (KOFT / KOFT-MHT): Building on classical Kalman filtering, we introduced trackers that incorporate optical flow as a second sensor. It improves motion modeling by better capturing sudden deformations in the tissue. We then extended this approach into a fully Bayesian Multiple Hypothesis Tracking (MHT) framework, which integrates priors on target dynamics and detection statistics. This probabilistic formalism further increases robustness to missed and false detections, significantly improving accuracy compared to widely used alternatives.

(3) Tracklet stitching with visual features (VEMC2): In calcium imaging, neurons are only detectable when active, leading to fragmented trajectories (tracklets). To address this, we proposed a stitching framework combining positional distances with self-supervised visual features extracted from fluorescence patches. By training a convolutional network though contrastive self-supervised learning, our method learns discriminative representations that help distinguish between neurons. A hybrid cost combining motion, position, and visual cues is optimized through a linear assignment problem, reducing stitching errors by a factor of two compared to previous approaches.

The methods were extensively validated on both SINETRA-generated data and real fluorescence recordings of *Hydra vulgaris*. Results demonstrate substantial improvements in tracking accuracy, robustness to noise, and continuity of reconstructed neuronal trajectories.

Beyond *Hydra vulgaris*, the proposed approaches are applicable to other small model organ-

isms such as *Caenorhabditis elegans* or zebrafish, and more generally to any biological system where dense, deformable, and intermittently detectable targets must be tracked over time. Moreover, we successfully adapted our method for cell tracking, where a key challenge is to detect and handle mitosis events. In fact, our method ranked first on the Cell Linking Benchmark[1] of the Cell Tracking Challenge, outperforming supervised approaches for this task.

This thesis contributes both methodological advances and practical tools to the tracking community, paving the way for more reliable analysis of biological videos. In particular, it enables efficient tracking of neuronal activity, providing new opportunities to study neuronal function in behaving animals.

Keywords: Multiple Object Tracking, Optimization, Bayesian models, Optical flow, Machine learning, Fluorescence imaging, Calcium imaging, *Hydra vulgaris*.

[1] `https://celltrackingchallenge.net/latest-clb-results/`

Résumé: Pour comprendre l'activité neuronale chez les animaux en activité, il est nécessaire de suivre avec précision de grandes populations neuronales au fil du temps. Dans les petits organismes modèles tels que *Hydra vulgaris*, l'imagerie calcique permet d'observer et suivre simultanément des centaines de neurones. Cependant, le suivi fiable reste un défi majeur en raison de plusieurs facteurs : fortes déformations corporelles, populations neuronales denses, bruit d'imagerie et visibilité intermittente des neurones dans les indicateurs calciques (GCaMP). Les algorithmes actuels échouent souvent dans ces conditions, produisant des trajectoires fragmentées ou des associations incorrectes qui compromettent l'interprétation biologique.

L'objectif de cette thèse est de développer des outils informatiques robustes pour le suivi d'objets multiples (MOT) adaptés à ces défis biologiques. Nos contributions s'articulent autour de trois axes complémentaires :

(1) Cadre de simulation (SINETRA) : afin de pallier le manque de données expérimentales annotées, nous avons développé SINETRA, un simulateur polyvalent de neurones intégrées dans des tissus déformables. Il génère des vidéos fluorescentes 2D/3D réalistes avec des annotations de référence, reproduisant des mouvement et des bruits similaires à ceux observés dans l'imagerie neuronal de l'Hydre. Ce benchmark synthétique permet une évaluation et une comparaison objectives des algorithmes de suivi.

(2) Suivi avec des modèles de mouvement améliorés par flux optique (KOFT / KOFT-MHT) : en nous appuyant sur les filtres de Kalman, nous avons introduit des trackers qui intègrent le flux optique comme un deuxième capteur. Ces méthodes améliorent le suivi en capturant mieux les déformations soudaines des tissus. Nous avons ensuite étendu cette approche à un cadre de Multiple Hypothesis Tracking (MHT) entièrement Bayésien, qui intègre des a priori sur la dynamique des cibles et les statistiques de détection. Ce formalisme probabiliste augmente encore la robustesse face aux détections manquées et fausses, améliorant considérablement la précision par rapport à l'état de l'art.

(3) Raccordement de tracklets avec des caractéristiques visuelles (VEMC2) : dans l'imagerie calcique, les neurones ne sont détectables que lorsqu'ils sont actifs, ce qui conduit à des trajectoires fragmentées (tracklets). Pour remédier à cela, nous avons proposé une méthode de raccordement de tracklets combinant les distances positionnelles avec des caractéristiques visuelles auto-supervisées extraites de patchs fluorescents. En entraînant un réseau convolutif par apprentissage contrastif, notre méthode apprend des représentations discriminantes qui aident à distinguer les neurones. Le raccordement est résolu via un problème d'assignation linéaire minimisant un coût hybride positionel et visuel, réduisant les erreurs de raccordement d'un facteur deux par rapport aux approches précédentes.

Les méthodes ont été largement validées à la fois sur des données générées par SINETRA et sur des enregistrements réels de fluorescence d'*Hydra vulgaris*. Les résultats démontrent des améliorations en termes de précision de suivi, de robustesse face au bruit et de continuité des trajectoires neuronales reconstruites.

Au-delà de l'Hydre, les approches proposées sont applicables à d'autres petits organismes modèles tels que *Caenorhabditis elegans* ou les poissons zèbres, et plus généralement à tout système biologique où des cibles denses, déformables et detectables par intermittence doivent être suivies dans le temps. De plus, nous avons adapté avec succès notre méthode au suivi cellulaire, où l'un des principaux défis consiste à détecter et à gérer les événements de mitose. Notre méthode s'est classée première au Cell Linking Benchmark, surpassant les approches supervisées pour cette tâche.

Cette thèse apporte à la fois des avancées méthodologiques et des outils pratiques à la communauté du suivi d'objets, ouvrant la voie à une analyse plus fiable des vidéos biologiques. Elle permet notamment un suivi efficace de l'activité neuronale, offrant ainsi de nouvelles possibilités pour étudier le fonctionnement neuronal chez les animaux en activité.

Mots-clés: Suivi d'objets multiples, Optimisation, Modèles Bayésiens, Flux optique, Apprentissage machine, Imagerie de fluorescence, Imagerie calcique, *Hydra vulgaris*.

Contents

Contents **ix**

List of Figures **xiii**

List of Tables **xiv**

Nomenclature and Abbreviations **xv**

I Introduction **1**
I.1 Funding and Collaborative Context . . . 1
I.2 Motivation . . . 1
I.2.1 *Hydra vulgaris*: an animal model . . . 2
I.2.1.1 Biology and Regeneration . . . 2
I.2.1.2 Neural System and Behavior . . . 2
I.2.2 Imaging Hydra's neurons . . . 5
I.2.2.1 Fluorescence Imaging . . . 5
I.2.2.2 Neuronal Activity Imaging . . . 6
I.2.2.3 Hydra example . . . 6
I.2.3 Experimental Data . . . 8
I.2.4 Analysis Pipeline . . . 9
I.2.4.1 Neuron Tracking . . . 10
I.2.4.2 Spike Extraction . . . 11
I.2.4.3 Ensemble Identification . . . 13
I.2.4.4 Behavioral Correlation . . . 13
I.3 Goals . . . 14
I.4 Challenges and Contributions . . . 15
I.4.1 Lack of Annotated Tracking Data . . . 16
I.4.1.1 Data for Supervised Learning . . . 16
I.4.1.2 Data for Quantitative Evaluation . . . 17
I.4.2 Tracking Challenges in Hydra . . . 17
I.4.2.1 Signal-to-Noise Ratio . . . 17
I.4.2.2 Neuron Density . . . 17
I.4.2.3 Animal Motion . . . 18
I.4.2.4 Intermittent Visibility . . . 18
I.4.3 Contributions & Publications . . . 18
I.4.3.1 Publications . . . 19
I.4.3.2 Manuscript Outline . . . 19

II Multiple Object Tracking: State of the Art **21**
II.1 Tracking-By-Detection framework . . . 22

II.1.1 Formalism . . . 22
II.1.1.1 Detection . . . 22
II.1.1.2 Linking . . . 23
II.1.1.3 State Estimation . . . 26
II.1.2 Detection Approaches . . . 27
II.1.2.1 Signal Enhancement and Thresholding . . . 27
II.1.2.2 Model-Based . . . 28
II.1.2.3 Deep Learning . . . 29
II.1.3 Linking approaches . . . 30
II.1.3.1 Graph-based . . . 30
II.1.3.2 Viterbi Track Linking . . . 35
II.1.3.3 Online linking . . . 37
II.1.3.4 Multiple-Hypothesis Tracking (MHT) . . . 43
II.2 Tracking-By-Regression . . . 48
II.2.1 Active Contours and Deformable Models (Snakes) . . . 48
II.2.2 Optical Flow-Based Tracking . . . 48
II.2.3 Feature-Based Registration . . . 49
II.2.4 Deep Regression for Multi-Object Tracking . . . 49
II.3 Bayesian Framework . . . 50
II.3.1 Single-Target State–Space Filtering . . . 50
II.3.1.1 Kalman filtering . . . 51
II.3.1.2 Biological Modeling Example . . . 52
II.3.1.3 Non-linear filtering . . . 53
II.3.1.4 Particle filtering . . . 54
II.3.1.5 Interactive Multiple Models (IMM) . . . 54
II.3.1.6 State-Space Smoothing . . . 56
II.3.2 Bayesian Multiple Object Tracking . . . 58
II.3.3 Random Finite Sets (RFS) . . . 60
II.3.3.1 Probability Hypothesis Density (PHD) . . . 61
II.3.3.2 Generalized Labeled Multi-Bernoulli (GLMB) . . . 61
II.4 MOT for Hydra's Neurons . . . 62

III SINETRA: a Synthetic Dataset for Evaluating Neuron Tracking 67
III.1 Motivation . . . 67
III.2 Method . . . 68
III.2.1 Modeling Fluorescent Targets and Background . . . 68
III.2.1.1 Targets . . . 68
III.2.1.2 Background & Noise . . . 69
III.2.1.3 Intensity Variations . . . 70
III.2.1.4 Shape Evolution . . . 70
III.2.2 Motion Modeling . . . 70
III.2.2.1 Optical Flow Motion . . . 70
III.2.2.2 Springs Motion . . . 71
III.2.2.3 Damped Harmonic Oscillators . . . 72
III.3 Evaluation and Results . . . 72
III.3.1 Evaluation . . . 72
III.3.2 Evaluated algorithms . . . 73
III.3.3 Results . . . 74
III.4 Discussion . . . 76

IV KOFT: Exploiting Optical Flow in Kalman-Based Tracking 79
IV.1 Motivation . . . 79
IV.2 Method . . . 80
IV.2.1 State-Space Modeling . . . 81
IV.2.2 Kalman Filtering with Dual Updates . . . 82
IV.2.3 Measurement and Process Models . . . 83
IV.2.4 Track-to-Detection Association . . . 84
IV.2.5 Optical Flow-Based Velocity Sensor . . . 84
IV.2.6 Track Creation and Termination . . . 85
IV.3 Results . . . 85
IV.3.1 Experimental Data . . . 85
IV.3.1.1 Synthetic Data . . . 85
IV.3.1.2 Hydra Vulgaris Data . . . 86
IV.3.2 Optical Flow Selection . . . 87
IV.3.3 Tracking . . . 89
IV.3.3.1 Experimental Setup . . . 89
IV.3.3.2 Tracking Performance on Synthetic Videos . . . 90
IV.3.3.3 Tracking Neurons in Hydra Vulgaris . . . 93
IV.3.4 Impact of imaging noise on optical flow and KOFT accuracy . . . 93
IV.3.5 Sensor Independence . . . 96
IV.4 Discussion . . . 97

V KOFT-MHT: Bayesian and Multiple Hypothesis Tracking Framework 99
V.1 Introduction . . . 99
V.2 Bayesian Multiple Object Tracking . . . 100
V.2.1 Notation . . . 102
V.2.2 Posterior Factorization . . . 105
V.2.3 MHT Solution . . . 107
V.2.3.1 Hypotheses Generation . . . 109
V.2.3.2 Cost Minimization . . . 110
V.2.3.3 Hypotheses Pruning . . . 110
V.2.4 Parameter Estimation . . . 111
V.2.5 State-Space Filtering . . . 114
V.2.5.1 Filtering parameters . . . 115
V.2.5.2 Initialization . . . 115
V.2.5.3 Covariance Growth . . . 115
V.2.6 State-Space Smoothing . . . 116
V.3 Experiments and Results . . . 117
V.3.1 Tracking Setup . . . 117
V.3.1.1 Parameters . . . 118
V.3.2 Tracking Results . . . 118
V.3.3 Depth impact . . . 120
V.3.4 Parameter Sensitivity . . . 121
V.4 Discussion . . . 122

VI VEMC2: Tracklet Stitching with both Visual and Motion Cues 125
VI.1 Motivation . . . 125
VI.2 Method . . . 126
VI.2.1 Tracklets . . . 126
VI.2.2 Tracklet Stitching . . . 127

VI.2.2.1 Positional cost . . . 127
VI.2.2.2 Visual cost . . . 127
VI.2.2.3 Temporal masking . . . 128
VI.2.2.4 Final cost matrix . . . 128
VI.2.2.5 Tracklet Stitching Evaluation . . . 128
VI.2.3 Costs mixture weights learning . . . 129
VI.2.3.1 Weights Learning Evaluation . . . 130
VI.2.4 Learning visual features . . . 130
VI.3 Results . . . 131
VI.3.1 Visual features learning . . . 131
VI.3.2 Tracklet stitching . . . 132
VI.3.3 Grid search versus Weights learning . . . 133
VI.4 Discussion . . . 133

VII Conclusion and Perspectives **135**
VII.1 Applications . . . 135
VII.1.1 Hydra Vulgaris . . . 135
VII.1.2 Cell Tracking Challenge . . . 138
VII.1.2.1 Cell Linking Benchmark . . . 138
VII.1.2.2 KOFT Adaptation . . . 138
VII.1.2.3 Parameter Estimation . . . 139
VII.1.2.4 Results . . . 139
VII.1.3 Broader Biological Applications . . . 140
VII.2 Thesis Summary . . . 141
VII.3 Limitations and Perspectives . . . 143

A Appendix: Additional results **147**
A.1 KOFT versus Warp-Then-Track . . . 147
A.2 MHT - Impact of the Motion Model . . . 148

Bibliography **151**

List of Figures

I.1 *Hydra vulgaris* . . . 2
I.2 Hydra anatomy . . . 3
I.3 Hydra behaviors . . . 3
I.4 Neuronal ensembles in Hydra . . . 4
I.5 Fluorescence imaging . . . 5
I.6 GECI signal vs neuronal voltage . . . 7
I.7 Hydra GCaMP . . . 7
I.8 GCaMP versus TdTomato fluorescence imaging . . . 8
I.9 Neuron tracking pipeline . . . 10
I.10 Neuron detection . . . 11
I.11 Spike extraction . . . 12
I.12 Behavior tracking . . . 14
I.13 Somersaults vs from neuronal activation . . . 15
I.14 Cell Tracking Challenge . . . 16

II.1 Illustration of the tracking-by-detection paradigm . . . 22
II.2 Illustration of track-oriented and measurement-oriented linking formalisms . . . 25
II.3 Graph-based linking approach . . . 31
II.4 Online linking algorithms . . . 38
II.5 Linear Assignment problem (LAP) . . . 40
II.6 Multiple Hypothesis Tracking (MHT) . . . 44
II.7 Hidden Markov model for tracks . . . 50
II.8 Tracking errors on Hydra . . . 64

III.1 Synthetic image simulator (Hydra flow) . . . 68
III.2 Synthetic image simulator (Springs 2D/3D) . . . 71
III.3 Temporal projection of tracks . . . 75

IV.1 KOFT overview . . . 81
IV.2 SINETRA synthetic images . . . 86
IV.3 Optical flow errors . . . 88
IV.4 Tracking performance with respect to detection accuracy . . . 91
IV.5 Temporal projection of tracks . . . 92
IV.6 Tracking performance on *Hydra vulgaris* . . . 94
IV.7 Impact of imaging noise on Farneback and KOFT . . . 95
IV.8 Error distributions for the two sensors . . . 96

V.1 Illustration of the Bayesian tracking framework formalism . . . 103
V.2 Multiple Hypothesis Tracking (MHT) . . . 108
V.3 Tracking performance with respect to detection accuracy . . . 119
V.4 Tracking performance with respect to tree depth . . . 120

V.5 KOFT-MHT sensitivity to parameters . . . 121

VI.1 Overview of the tracking pipeline with visual tracklet-stitching . . . 126
VI.2 Tracklet stitching results on Hydra . . . 133

VII.1 Remaining tracking errors on Hydra . . . 137
VII.2 Long-term tracking of TdTomato neurons over 1000 frames . . . 137
VII.3 Examples of broader biological applications . . . 141

A.1 Tracking performance with respect to detection accuracy and motion models . . . 149

List of Tables

III.1 SINETRA benchmark . . . 74

IV.1 Optical flow benchmark . . . 87
IV.2 Tracking performance on SINETRA . . . 91

V.2 MHT parameters . . . 111
V.3 Summary of experimental settings . . . 118
V.4 Tracking performance on SINETRA . . . 119

VI.1 Hydra data and self-supervised visual training results . . . 132
VI.2 Tracklet links retrieval . . . 132

VII.1 Cell Linking Benchmark results . . . 140

A.1 Warp-Then-Track vs KOFT . . . 147

Nomenclature and Abbreviations

Symbol	Description
t	Temporal index
i	Target index
j	Detections index
T	Number of frames in the video
d	Spatial dimension of images (Usually, $d \in \{2, 3\}$ for 2D and 3D images)
S	$\in \mathbb{N}^d$, spatial sizes (in pixels) of images along each dimension
c	Number of channels of images
x, y, z	Spatial coordinates (pixel)
$\dot{x}, \dot{y}, \dot{z}$	Temporal derivative (velocity) of spatial coordinates (pixel by frame)
$\ddot{x}, \ddot{y}, \ddot{z}$	Temporal double derivative (acceleration) of spatial coordinates (pixel by frame^2)
$\mathbf{w}$	$\in \mathbb{R}^d$, a point in the image space (pixel)
I	$: \mathbb{R}^d \to \mathbb{R}^c$, an image over the image space
$I(\mathbf{w})$	$\in \mathbb{R}^c$, intensity(ies) of image I evaluated on pixel $\mathbf{w}$. Extrapolated for non-integer pixels.
$I_{1:T}$	$= (I_1, \dots, I_T)$, a video of T images (frames)
N	Number of targets in the video
M_t	Number of detections on frame t
$d_{\mathbf{x}}$	Dimension of the state space used for targets
$d_{\mathbf{z}}$	Dimension of the measurement space
$\mathbf{x}_t^i$	$\in \mathbb{R}^{d_{\mathbf{x}}}$, state vector of the i^{th} target at time t. The i superscript may be dropped for single target systems
$\mathbf{z}_t^j$	$\in \mathbb{R}^{d_{\mathbf{z}}}$, measurement vector of the j^{th} detection at time t
X_t	$= (\mathbf{x}_t^1, \dots, \mathbf{x}_t^N)$, states of all the targets at time t
$X_{1:T}^i$	$= (\mathbf{x}_1^i, \dots, \mathbf{x}_T^i)$, all the states of the i^{th} target
$X_{1:T}$	$= (X_1, \dots, X_T)$, states of all the targets on all the frames
Z_t	$= (\mathbf{z}_t^1, \dots, \mathbf{z}_t^{M_t})$, all the detections measurements at time t
$Z_{1:T}$	$= (Z_1, \dots, Z_T)$, all the detections
J_t	$= [\![1,\, M_t]\!]$, detections indices at time t
$\mathsf{P}(S)$	$= \{s \mid s \subset S\}$, power set of a given ensemble S

$\Omega(T)$	$\bigotimes_{t=1}^{T} \mathsf{P}(J_t)$, set of all feasible associations between targets and detections
γ_t^i	$\subset J_t$, set of the associated detections indices to the i^{th} target at time t
$\gamma_{1:T}^i$	$\in \Omega(T)$, all the associations for the i^{th} target ("track"). Abbreviated γ^i.
$\Gamma_{1:T}$	$= (\gamma^1, \dots, \gamma^N) \subset \Omega(T)$, the set of tracks (track-oriented notations)
L_t^j	$\in [\![1, N]\!]$, labeling of the j^{th} detections
$L_t, L_{1:T}$	Sequence of labels over all the detections, and all the frames (measurement-oriented notations)
$\mathbf{C}_{ij}$	$\in \mathbb{R}^+$, assignment cost between track $\gamma_{1:t}^i$ and detection $\mathbf{z}_{t+1}^j$
η	$\in \mathbb{R}^+$, cost threshold above which assignments are not considered
$\mathbb{P}(\cdot), p(\cdot)$	Discrete probability and continue density distribution
$\mathcal{B}(\,\cdot\,; p)$	Bernoulli distribution with probability of success p
$\mathcal{B}(\,\cdot\,; p, n)$	Binomial distribution with n trials and probability of success p
$\mathcal{N}(\,\cdot\,; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$
$\mathcal{P}(\,\cdot\,; \lambda)$	Poisson distribution distribution with mean λ

Abbreviation	**Meaning**
KF	Kalman Filter
IMM	Interactive Multiple Model
MHT	Multiple Hypothesis Tracking
LAP	Linear Assignment Problem
MAP	Maximum A Posteriori / Multi-dimensional Assignment Problem
MILP	Mixed Integer Linear Programming

Chapter I

Introduction

I.1 Funding and Collaborative Context

This PhD project began in April 2022 and is supported by an *Agence Nationale de la Recherche* (ANR-21-CE45-0020-01 REBIRTH) grant awarded to Thibault Lagache. The work is carried out jointly at the **Biological Image Analysis (BIA)** laboratory of the *Institut Pasteur* in Paris, headed by Jean-Christophe Olivo-Marin, and the **IMAGES** team at *Télécom Paris*, led by Yann Gousseau. Day-to-day supervision is provided by Thibault Lagache and Jean-Christophe Olivo-Marin (Institut Pasteur) together with Alasdair Newson and Elsa Angelini (Télécom Paris).

A close scientific collaboration has been established with Rafael Yuste's laboratory at Columbia University in New York. In this project, the Yuste lab focuses on the biological and neuroscientific investigation of the freshwater cnidarian *Hydra vulgaris*, the model organism around which the majority of this thesis is based. Their group supplies most of the experimental data (microscope images & video), while our group develops computer vision tools and methods for quantitative analysis of these images.

In particular, this PhD project was centered on the development of robust tracking algorithms for the neurons of the *Hydra vulgaris* in fluorescence images. A year ago, Maria Manich joined our Hydra project in the BIA lab to take charge of breeding *Hydra vulgaris* and conducting experimental recordings directly within the lab at Institut Pasteur. This development marked a key transition, enabling the team to complement its computational work with in-house biological experimentation and to begin addressing more biologically-oriented research questions. Nonetheless, in the scope of this PhD, we focused on tracking algorithms for microscopy images.

I.2 Motivation

Understanding how neural circuits encode and drive behavior is a central challenge in neuroscience. In our lab at Institut Pasteur, we address this question through an interdisciplinary project centered on the freshwater cnidarian *Hydra vulgaris* using modern imaging techniques and computational methods to observe and analyze the animal's entire nervous system in action.

By simultaneously monitoring many—or eventually all—nerve cells and the animal's behavior, we seek to decipher how patterns of neural activity map to specific motor outputs. A key step toward this goal is the robust tracking of hundreds to thousands of fluorescent neurons while the live animal freely behaves. In practice, standard Multiple Object Tracking (MOT)

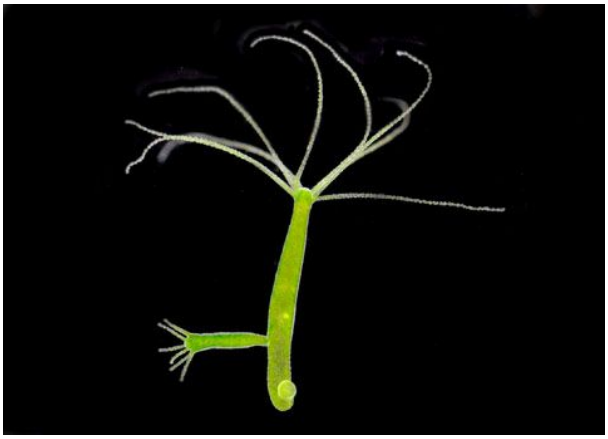


Figure I.1: *Hydra vulgaris*. Adult polyps are 1–10 mm long and possess 4–10 poison-tipped tentacles used for prey capture and locomotion. The photograph illustrates asexual reproduction by *budding*, where a new individual emerges from the parent's body column.

algorithms fail on Hydra's neurons because of data-specific challenges that will be detailed in the following of this introduction. To this end, the present thesis aims to provide more robust MOT algorithms. Although our focus is mainly Hydra neurons, the developed methods are intended to be generic and transferable to other biological targets (cells, molecules, pathogens).

Figure I.1 illustrates the Hydra animal at the heart of this thesis. The remainder of this section introduces *Hydra vulgaris* as a model organism for neuroscience, then, briefly reviews fluorescence imaging techniques used to record neural activity and finally, explains why these topics motivate the computational tracking algorithms that we developed in this thesis.

I.2.1 *Hydra vulgaris*: an animal model

I.2.1.1 Biology and Regeneration

Hydra vulgaris is a small freshwater polyp belonging to the Cnidarian. Its tubular body (Figure I.2.A) has a head (hypostome) at one end, surrounded by tentacles and an oral opening, and a basal disc (foot) at the other, which allows it to anchor to surfaces. It consists of two primary tissue layers: an outer ectoderm and an inner endoderm, separated by a non-cellular mesoglea (Figure I.2B). Despite this structural simplicity, the Hydra has been studied for more than 300 years due to its ease of laboratory cultivation, transparency, and exceptional regenerative capacity [2–4].

It is capable of both sexual and asexual reproduction. The latter occurs through *budding*, as illustrated in Figure I.1. Remarkably, Hydra does not undergo senescence: it continuously regenerates its tissues through three distinct stem cell lineages (Figure I.2C), allowing full body regeneration even from small tissue fragments [4–7].

I.2.1.2 Neural System and Behavior

Hydra's nervous system is composed of a decentralized nerve net distributed across both its ectodermal and endodermal tissue layers (Figure I.2). Unlike more evolved nervous systems, Hydra lacks a central brain. Instead, sensory neurons, sensitive to mechanical and light stimuli, are scattered throughout the body and tentacles. These are connected through ganglion neurons to muscles allowing to trigger coordinated muscular responses.

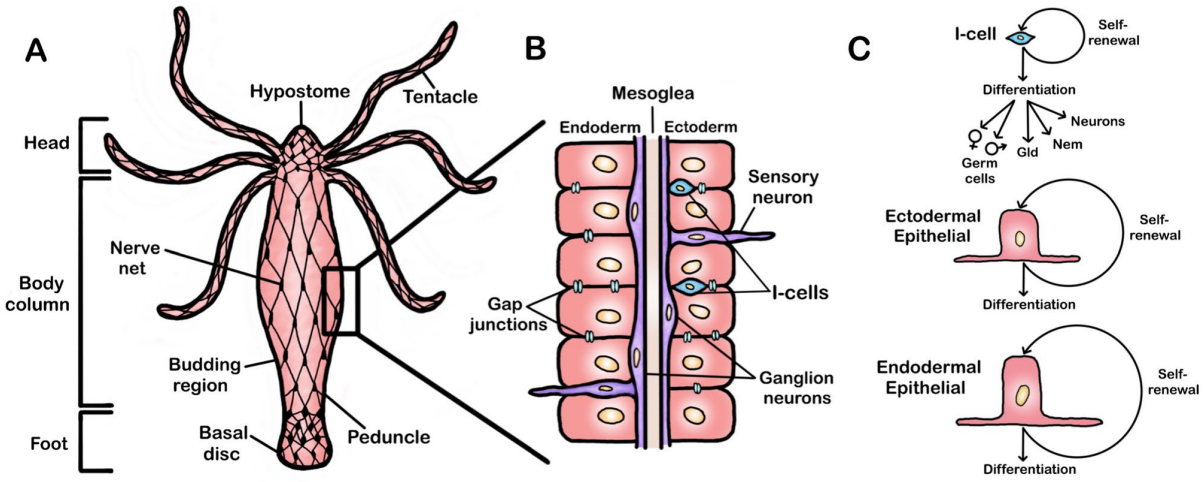


Figure I.2: Hydra anatomy from [1]. **(A)** The basic body plan of Hydra. **(B)** The two tissue layers of Hydra are depicted with endodermal epithelial cells on the interior, ectodermal epithelial cells on the exterior, and an acellular mesoglea in between. Ganglion neurons are found at the base of the epithelio-muscular cells while sensory neurons protrude from both the endoderm and ectoderm. Interstitial-cells (I-cells) are located between ectodermal epithelial cells. **(C)** The three stem cell lineages of Hydra are shown with the unipotent endodermal and ectodermal epithelial cells and multi-potent I-cells that produce four cell types: germ cells, gland cells (Gld), nematocytes (Nem), and neurons.

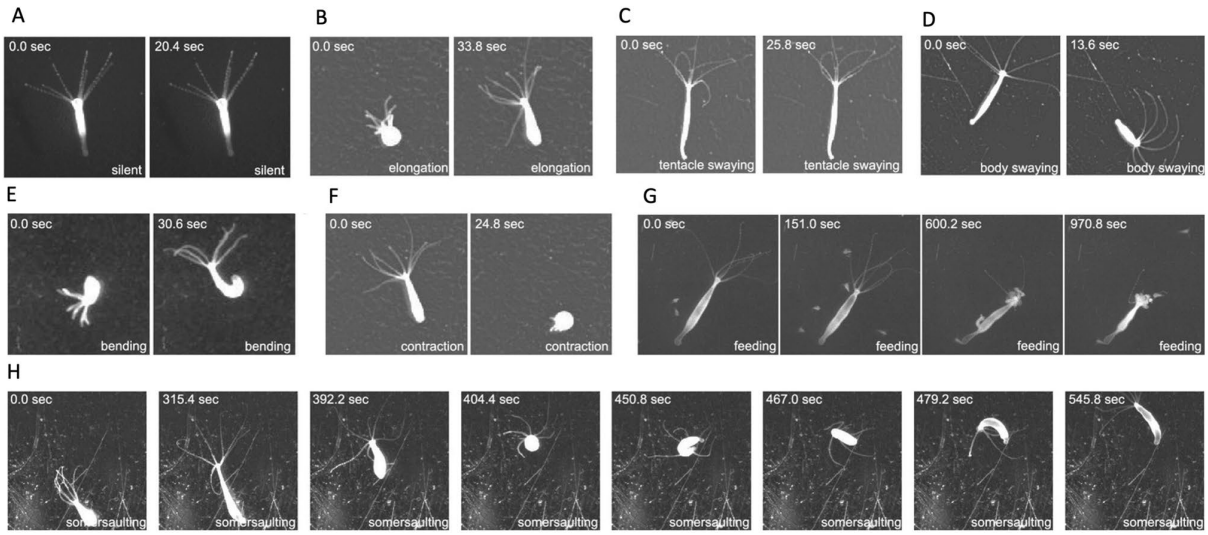


Figure I.3: Hydra behaviors from [1,8]. **(A)** Resting state. **(B–F)** Simple actions: elongation, tentacle swaying, body swaying, bending, full-body contraction. **(G–H)** Complex behaviors: feeding and somersaulting.

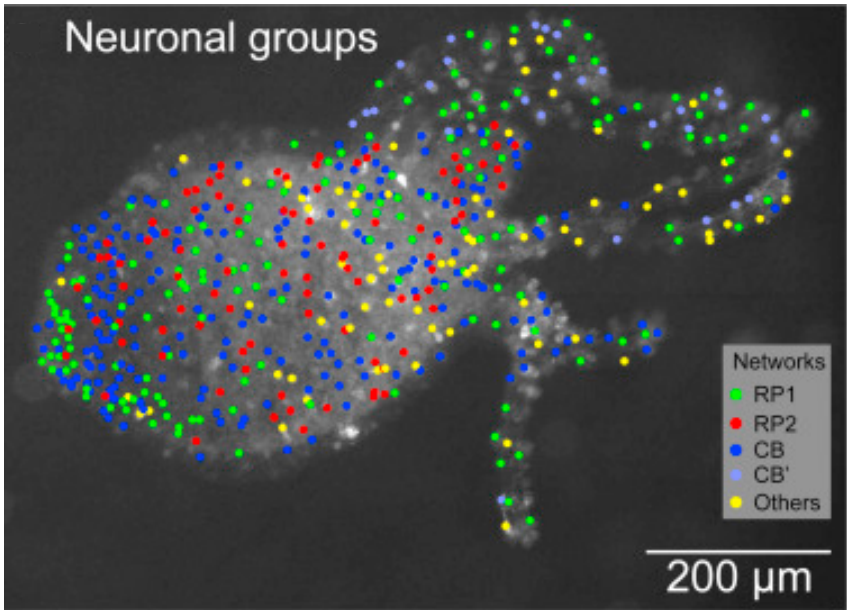


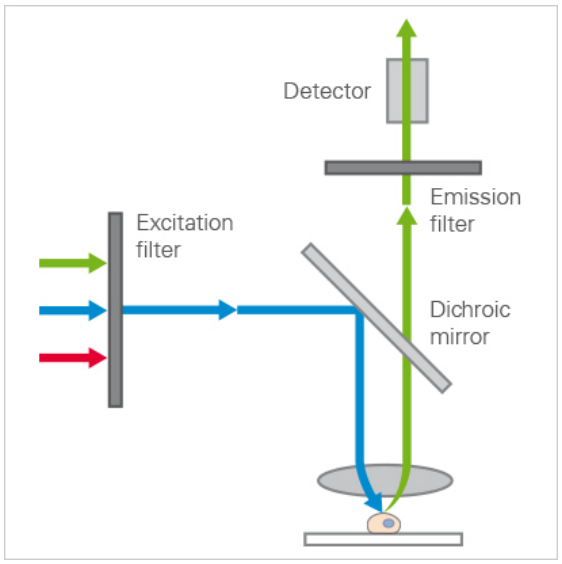
Figure I.4: Topographical distribution of neuronal ensembles in Hydra, from [10]. Neurons are grouped into functional ensembles: rhythmic potential 1 (RP1, green), rhythmic potential 2 (RP2, red), contraction burst neurons (CB, dark/light blue), and other uncategorized neurons (yellow). CB' indicates neurons of the tentacles that did not fire during the two CB events of this time window but fired during another CB event.

Hydra possesses only a few thousand neurons from ∼10 different cell types [9], positioning it among the simplest nerve nets in the animal kingdom. Nevertheless, it displays a rich behavioral repertoire including spontaneous and stimulus-driven motor actions such as tentacle swaying, elongation, contractions, and complex actions like feeding and somersaulting (Figure I.3) [1, 8]. As such, Hydra serves as an intermediate model between the well-studied *Caenorhabditis elegans* with its 302 neurons, and more complex vertebrates such as mice ($\sim 7\times10^7$ neurons) and humans ($\sim 8 \times 10^{10}$ neurons).

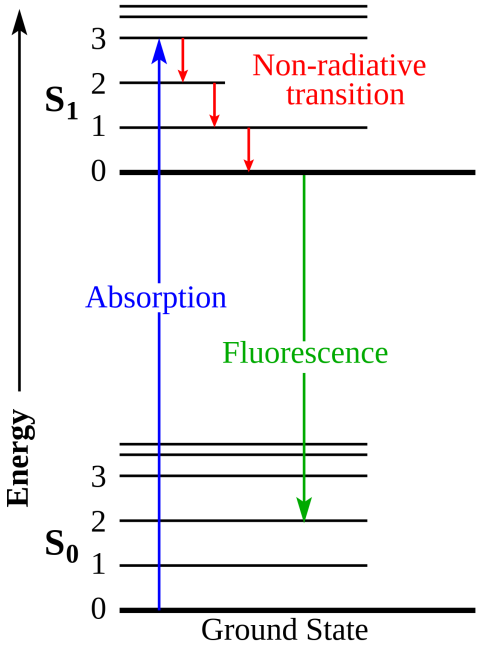
Recent discoveries have shown that Hydra's neurons do not act independently but are organized into functional ensembles—groups of co-active neurons that fire together to control specific behaviors. These ensembles form spatially-distributed but functionally distinct units within the nerve net. For example, the **CB** (contraction burst) ensemble mediates whole-body contractions, while **RP1** (rhythmic potential 1), exhibits spontaneous oscillatory activity in the absence of sensory input and accelerates their rhythm under light stimuli and elongation of the animal [10].

Neuronal functional ensembles are spatially intermixed (See Figure I.4) but do not overlap in their functions and activations, forming non-overlapping subnetworks within the nerve net. The relatively low number of neurons and their functional organization into ensembles make Hydra a uniquely tractable system for investigating the neural code—*i.e.* how spatially and temporally distributed patterns of neural activity encode and control behaviors.

As neurons are sparsely distributed in the body (Figure I.4), recent advances in optical microscopy and genetic engineering allow whole-body and cellular-resolution imaging of neural activity in Hydra using genetically-encoded markers. It is therefore possible to visually monitor the activity of hundreds to thousands of neurons in real time, capturing their dynamics during natural behavior. This will be discussed in the following section.



(a) Wide-field fluorescence microscope. Excitation light from a laser (blue) illuminates the sample; emitted photons (green) are collected by the objective and imaged on an camera.



(b) Fluorophore excitation/emission. Emitted photons have lower energy (longer wavelength) than absorbed photons.

Figure I.5: Principles of fluorescence microscopy. Fluorophores absorbs light of a specific excitation wavelength, and subsequently emits a photon with a longer wavelength, which is isolated and recorded to form an image.

I.2.2 Imaging Hydra's neurons

I.2.2.1 Fluorescence Imaging

Fluorescence microscopy is a powerful, non-invasive method to visualize specific biological processes *in vivo*. It relies on fluorophores (*e.g.* fluorescent protein) that absorb excitation light and re-emit it at a longer wavelength (Figure I.5). The Green Fluorescent Protein (GFP) [11,12] is a famous example, with many derivatives used to target specific cells or molecules.

The main benefit of fluorescence is that it enables specific cells or structures to be targeted with fluorophores. Marked structures are fluorescent when illuminated, while the other parts of the sample (background) are not, directly improving contrast. Despite being efficient with thin samples, when directly applied to thick and living samples, the background may express auto-fluorescence, and light scatters in tissues, which reduces the Signal to Noise Ratio (SNR) and spatial resolution. To overcome these limitations, confocal microscopy [13] or two-photons microscopy [14] improves over wide-field microscopy by restricting the photo-stimulation to a precise focal plan.

In addition, cells and fluorophores suffers from photostability problems: cells are damaged by the emitted radiations, which can lead to cell and organism death (phototoxicity), and fluorophores can only withstand a finite number of absorption/emission cycles before switching to a permanent dark state (photobleaching). Advanced microscopy modalities—confocal spinning-disk, structured illumination, two-photon, light sheet—optimize the trade-off between spatial resolution, light penetration depth and photostability [15].

I.2.2.2 Neuronal Activity Imaging

Electrophysiology (patch clamp, multi-electrode arrays) has been the gold standard for single-neuron voltage recordings for decades. Nonetheless, such methods are invasive and limited in scalability, recording the activity of only a few neurons (2–3 with patch-clamp, up to a few dozen with multi-electrode arrays) simultaneously [16–18], which restricts their applicability for behaving animals. Fluorescent microscopy circumvent these issues by converting electrical activity into an optical signal through fluorophores expressed in the nerve cells (Figure I.6. Two main families of neural fluorophores exist:

Calcium Indicators (GECIs) Genetically-encoded calcium indicators (GECIs) are fluorophores that couple intracellular calcium concentration to fluorescence. A famous GECI protein is the GFP-Calmodulin-M13Peptide (GCaMP) [19], which we used in this project.

These calcium indicators are sensitive to changes in calcium concentrations in neurons induced by a membrane depolarization and action potentials (Figure I.6b). For GCaMP, upon calcium influx, calcium ions (Ca^{2+}) bind to the GCaMP proteins, which become fluorescent (blue absorption at 480nm and green emission at 510nm) [19].

However, calcium signals are only indirect proxies of membrane voltage creating some discrepancies between the GECI and the real voltage signal: GECI signal misses sub-threshold depolarization, is polluted by external cellular processes that produce notable changes in calcium ion concentration, and suffers from a poor (slow) temporal resolution [20].

Voltage Indicators (GEVIs) Genetically encoded voltage indicators (GEVIs) overcome some of these issues by providing a direct readout of membrane voltage, thus improving temporal fidelity (Figure I.6a) [21]. However, they suffer from lower photon efficiency (and hence poorer SNR) compared to calcium indicators [20]. Moreover, at the beginning of this PhD project, model systems such as *Hydra* had not yet been engineered to express these indicators.

I.2.2.3 Hydra example

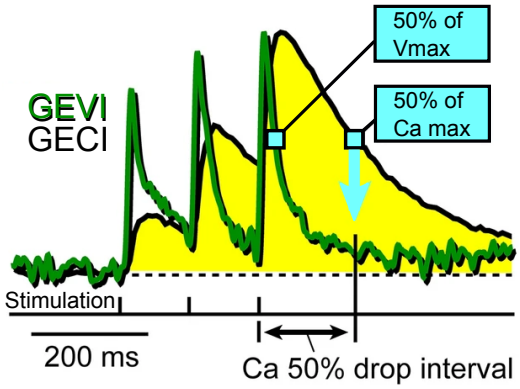
Hydra is an ideal subject for calcium imaging: the animal is small and transparent, neurons are spatially distributed and separated from each other, and neuronal activity is action-potential based, slow, and sparse [10].

In that case, GECI with GCaMP allows retrieving neuronal spikes from the calcium trace (Figure I.6b). As such, in this thesis we relied primarily on GCaMP recordings for their higher signal SNR compared with GEVI.

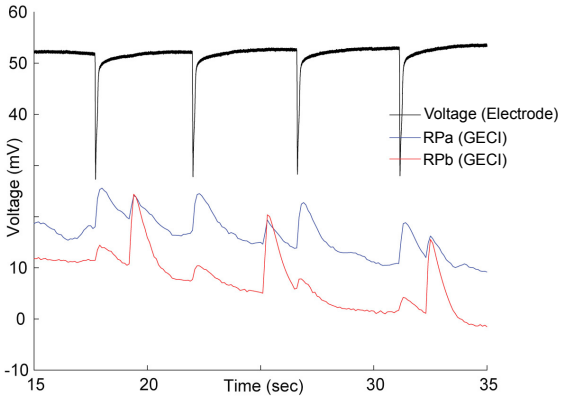
An example of Hydra imaged with GCaMP fluorescence is shown on Figure I.7. The full video can be retrieved on Dryad[1]. The whole body of the animal usually fits in the field of view, allowing the recording of several hundred ($\sim$ 800) neurons with the behavior of the animal. Videos are usually acquired at $\sim 10\,\mathrm{Hz}$ balancing temporal resolution and image quality. The animal is scanned using confocal microscopy in 2D. To prevent the animal going out of the focus plane, it is usually stuck between two plates, reducing slightly its behavioral capabilities.

In conclusion, *Hydra vulgaris* provides a powerful system to study whole-organism neural activity *in vivo*. The rich behavioral repertoire, small nervous system, and compatibility with live fluorescence imaging make it ideal for investigating the relationship between neural dynamics

[1] `https://datadryad.org/dataset/doi:10.5061/dryad.h9w0vt4q3` (G7_contrxn-1.avi)



(a) Neuronal fluorophores in mice, GECI vs GEVI from [20]. GECI (black) and GEVI (green) optical signals are superimposed on the same time scale. Turquoise points indicate when the GECI and GEVI signal amplitudes dropped down to 50% of their maxima. When GECI signal is at 50%, the GEVI trace is near the baseline (0–5%)



(b) GECI vs electrode in Hydra from [10]. Extracellular electrode placed close to a neuron from RP network (RPa) and far from a neuron from the other RP network (RPb). Accordingly, the spikes recorded in electrical activity match calcium spikes of neuron A (blue trace), but not neuron B (red trace). This also illustrates cross-contamination spikes between the fluorescence signals from neurons A and B, which can distinguished from true spikes thanks to their amplitude difference (large true spikes in neuron A induces small cross-contamination spikes in neuron B, and vice versa).

Figure I.6: Calcium-based fluorophores (GECI) vs voltage (Electrode or GEVI fluorophores)

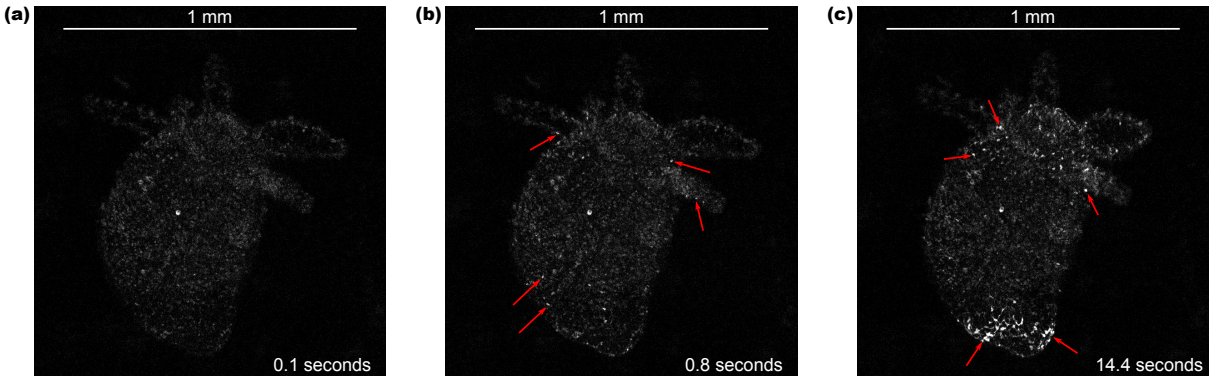


Figure I.7: Hydra imaged with GCaMP fluorescence. A spinning disk confocal microscope (10Hz) is used to image a GCaMP Hydra in 2D [22] **(a)** Neurons at resting state at the beginning of the video. **(b)** A neuron ensemble is firing at 0.8 seconds. Red arrows indicates some of these neurons. **(c)** Contraction burst (CB) ensemble fires (repeatedly), triggering a contraction of the whole animal in the following seconds (15s to 45s).

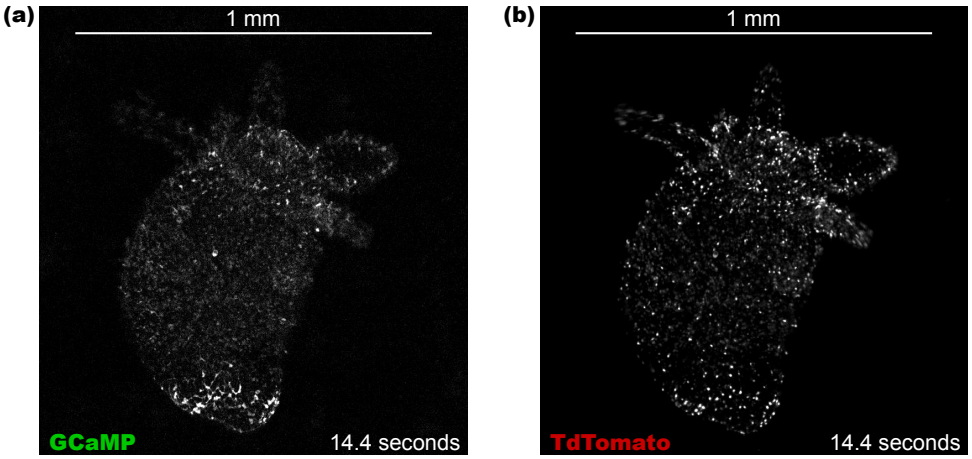


Figure I.8: GCaMP versus TdTomato fluorescence imaging. Our TRASE-IN [22] project uses transgenic Hydra specimens expressing both GCaMP (left), which reports neuronal activity, and TdTomato (right), which labels neuron nuclei independently of activity. The two signals are shown at $t = 14.4\,\mathrm{s}$, at the onset of a contraction. **(a)** GCaMP has activity-dependent signals therefore neurons are undetectable when inactive. **(b)** TdTomato offers a high SNR and stable visibility of neurons regardless of activity, aiding tracking and signal normalization.

and behavior. In the following sections, we first detail the experimental videos used in this project, and then discuss the automated analysis pipeline of such microscopy videos enabling large-scale analysis of neuronal activity over time.

I.2.3 Experimental Data

Throughout this thesis, we relied on experimental fluorescence videos of *Hydra vulgaris* acquired in collaboration with Rafael Yuste's laboratory at Columbia University. In their setup, the genetically-encoded calcium indicator GCaMP is used to report neuronal activity. An example of such a video is shown in Figure I.7. Recordings are typically performed with a spinning disk confocal microscope at 6× magnification and a frame rate of 10 Hz. Depending on the experimental protocol, acquisition spans from one minute to an hour, yielding sequences ranging from a few hundred to several thousand frames.

For instance, the datasets used in the TRASE-IN project [22] (see the following Section) consist of videos of 1000 frames (100 s) with a resolution of 1000×1000 pixels. Each sequence contains the full body of the animal with approximately 800 neurons to track. Individual neurons occupy a roughly circular region of radius 2–3 pixels.

In the TRASE-IN project, Alison Hanson (Yuste lab) cultivated double-transgenic Hydra expressing both GCaMP for calcium activity and TdTomato, fluorophore (emission in red) that marks neuron nuclei. An illustration is provided in Figure I.8, and the corresponding full-length videos are available on Dryad[2]). While GCaMP enables functional imaging of neuronal activity, its intermittent fluorescence leads to periods of undetectability when neurons are silent. In contrast, TdTomato provides a stable, activity-independent signal that is particularly valuable for tracking and intensity detrending. Imaging is performed with two excitation lasers tuned to the respective absorption wavelengths of GCaMP and TdTomato, and two synchronized detectors capture their emission spectra, producing two-channel fluorescence videos.

[2] `https://datadryad.org/dataset/doi:10.5061/dryad.h9w0vt4q3` (files `G7_contrxn-1.avi` and `tdt_contrxn-1.avi`

Normalization. Raw microscopy videos are not directly scaled to the standard intensity range $[\![0, 255]\!]$. For both visualization and downstream analyses (detection, tracking, optical flow), raw intensities I_{raw} are normalized on a per-pixel basis as

$$I(\mathbf{w}) = 255 \times \max\left(0, \min\left(1, \frac{I_{\text{raw}}(\mathbf{w}) - q(I, 0.01)}{q(I, 0.999) - q(I, 0.01)}\right)\right) \in [\![0, 255]\!]\,,$$

where $q(I, p)$ denotes the p-quantile of intensities, computed over the first 50 frames of the video.

Annotations. Annotations for Hydra neuronal imaging are scarce. Within TRASE-IN, we manually annotated 20 TdTomato images to train supervised segmentation and detection algorithms (see the following Section). To the best of our knowledge, no tracking annotations are available for Hydra neurons, with the sole exception of a partial 20 s GCaMP sequence annotated by C. Dupre [10], which has several limitations discussed in Chapter IV.3.1.2.

I.2.4 Analysis Pipeline

In this section, we present a general automated pipeline for the analysis of Hydra fluorescence videos, designed to extract individual neuronal activities and relate them to the animal's behavior. In particular, we will rely on our method TRASE-IN [22] that we developed at the beginning of this PhD project and neuronal functional ensemble analyses developed by a former PhD student in our lab at Institut Pasteur [23].

In our TRASE-IN pipeline, we rely on double-transgenic Hydra, expressing both GCaMP and TdTomato. However, cultivating animals that express both fluorophores is technically challenging. As such, most experiments only include GCaMP. Consequently, the pipeline described below focuses on a more general case in which only activity-dependent GCaMP images are available.

The full analysis pipeline can be summarized in four main stages:

1. **Neuron tracking:** Individual neurons are tracked across the video to extract the calcium signal of each neuron.

2. **Spike extraction:** Raw calcium traces (Figure I.6a) are denoised and deconvolved to infer binary activation spike.

3. **Ensemble identification:** Neurons are grouped into functional ensembles based on co-activity patterns.

4. **Behavioral correlation:** Ensemble activity is mapped to behavior to explore neural coding principles.

We emphasize that this PhD project focused on the neuron tracking stage, with the exception of the early TRASE-IN project, which also addressed spike extraction. We nevertheless describe the full analysis pipeline to place our work in context and to give the reader a comprehensive view of the complete project.

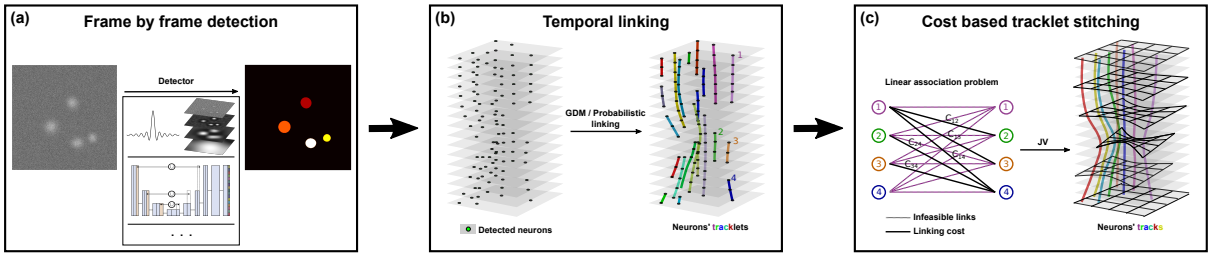


Figure I.9: Neuron tracking pipeline in Hydra. **(a)** Neurons are detected in each frame using either signal-enhancement and thresholding methods or deep learning. **(b)** Detections are linked across time using Bayesian optimization to form tracklets. **(c)** Tracklets are stitched into complete trajectories using spatiotemporal and appearance-based cues.

I.2.4.1 Neuron Tracking

Tracking individual neurons in GCaMP (or TdTomato) videos is a classic multiple object tracking (MOT) problem, which is treated in depth in Chapter II. We adopt the versatile tracking-by-detection paradigm.

GCaMP images suffer from intermittent detectability: when neurons are inactive, their signal often drops below detection threshold. Moreover, imaging artifacts such as noise, out-of-focus blur, and occlusions contribute to missed detections. As neurons may be undetectable for more than 100 consecutive frames, standard MOT algorithms tend to produce short, fragmented trajectories called *tracklets*. These partial tracks must be linked into full-length trajectories using a dedicated *tracklet stitching* step.

Our full tracking pipeline is thus divided into three stages (Figure I.9): **(1)** Detection of neurons in each frame; **(2)** Linking of detections into tracklets; **(3)** Stitching of tracklets into coherent long-range trajectories.

Detection See Chapter II.1.2 for a short review of detection and instance segmentation methods. Throughout this thesis, we usually rely on the wavelet-based spot detection method from [24], which is particularly effective for detecting small, round, bright targets such as neurons. In the TRASE-IN project, we found it achieved around 80% F1-score in our annotated TdTomato images and generalizes reasonably well to GCaMP, only requiring to adapt its main parameters-*i.e* the approximate object size (scale) and an intensity threshold. However, the wavelet approach struggles to resolve densely packed neurons, since it converts thresholded binary masks into discrete detections via connected components (two closely apposed neurons correspond to a single connected component).

In our TRASE-IN project, we overcome this by training a deep learning model (StarDist [25]) on our annotated TdTomato images. This approach achieved ∼90% F1-score and better separated adjacent nuclei (Figure I.10). While StarDist is more accurate, its generalization is limited. It performs poorly on GCaMP images or data acquired with different microscopes unless retrained, which requires time-consuming annotation. The wavelet-based method, though less precise, is more versatile and easier to adapt via simple parameter tuning. In this thesis, we assume reasonably accurate detections and focus our contributions on improving the other tracking stages.

Linking Various algorithms are available for linking detections into tracklets, as reviewed in Chapter II.1.3. Early in this project, we relied on eMHT [26], a multiple hypothesis tracking

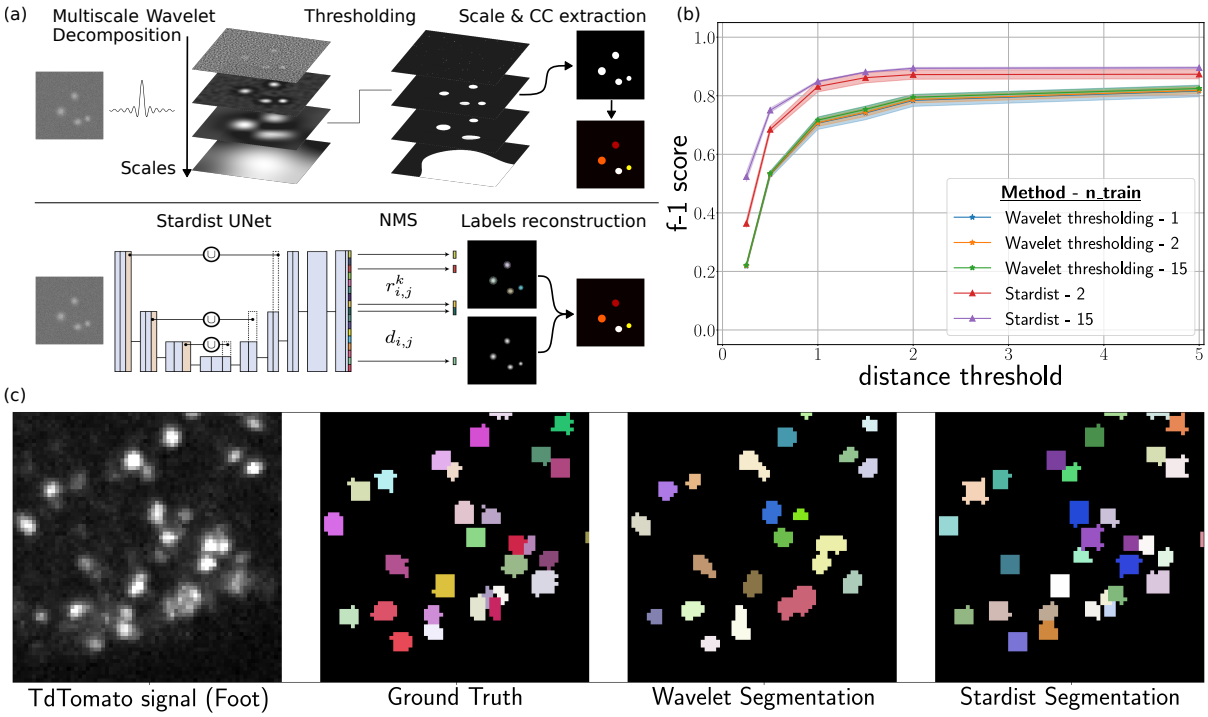


Figure I.10: Neuron detection in TdTomato from [22]. **(a)** Comparison of two detection algorithms: Top: wavelet-based multiscale denoising and thresholding [24] It uses connected component labeling (CC) to resolve discrete detections from the binary segmentation mask. Bottom: StarDist [25] neural network-based segmentation. Non Maximum Suppression (NMS) is used to filter out redundant predictions. **(b)** F1-score accuracy on test dataset of our TdTomato annotated images, comparing wavelet and deep learning for various training data sizes. For the wavelet spot detector, training images are used to find the optimal parameters by grid-search. **(c)** Example TdTomato image with ground truth annotations and instance segmentation from the wavelet spot detector and StarDist

(MHT) approach that frames linking as a Bayesian optimization problem. However, eMHT suffers from scalability and robustness issues in Hydra data (that will be discussed later on). These limitations motivated the development of new linking strategies, detailed in Chapters IV and V.

Stitching Due to prolonged undetectability of neurons in GCaMP images, linking often yields fragmented tracklets. These must be stitched into full trajectories. This step is typically formulated as a cost minimization problem over potential tracklet-to-tracklet associations. Cost functions include Euclidean distance [27] and motion-compensated metrics such as those used in EMC2 [28]. The Hungarian algorithm [29] or its faster Jonker-Volgenant (JV) implementation [30] are then used to find optimal links.

In TRASE-IN, we employed EMC2, a method previously developed in our lab. During this PhD, we further extended this approach by incorporating visual features into the stitching process, which is detailed in Chapter VI.

I.2.4.2 Spike Extraction

Following neuron tracking, spiking activity are extracted for individual neurons. To facilitate the analysis, short tracks are discarded, and long tracks are interpolated at missing positions

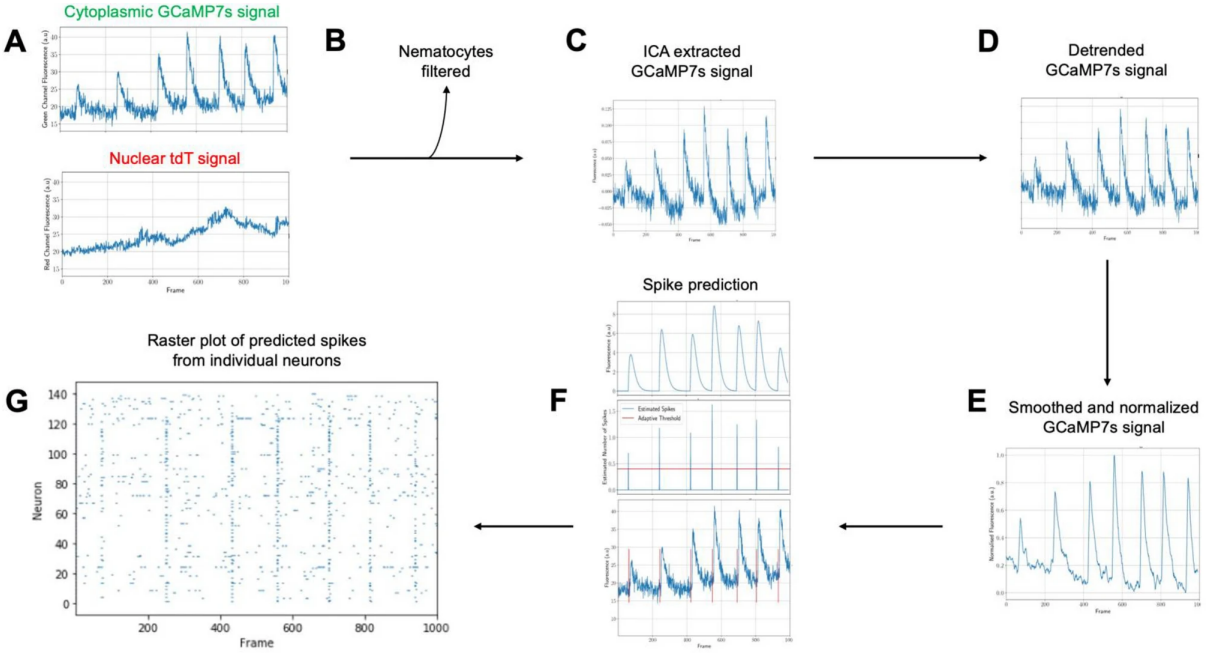


Figure I.11: Extraction of spikes from raw calcium signals in TRASE-IN [22]. **A** Raw GCaMP and TdTomato signals are extracted for each neuron. **B** Non-neuronal cells (e.g., nematocytes) are filtered out. **C** Independent Component Analysis (ICA) uses the TdTomato signal to remove non-calcium-related variations. **D**–**E** Signals are detrended and smoothed. **F** Spikes are inferred using the Foopsi algorithm [31] followed by adaptive thresholding. **G** Resulting binary spikes for each neuron across time.

over the entire video using the motion of neighboring tracks. Manual curation may be applied to remove erroneous or false-positive tracks, ensuring that a substantial number of neurons are accurately tracked throughout the recording.

For each tracked neuron, the raw calcium signal is extracted from the GCaMP images at the tracked positions (Figure I.11A). Specifically, we compute the mean GCaMP fluorescence intensity over the segmented neuron area provided by the tracking algorithm. If segmentation is unavailable for certain frames (due to missed detections or for tracking methods that do not include a segmentation stage), a user-defined circular ROI centered on the track position can be used instead.

Let y_t denote the noisy raw fluorescence signal extracted for a given neuron at frame t. This signal is usually modeled as [31, 32]:

$$y_t = a(c_t + b) + b_t + \epsilon_t, \tag{I.1}$$

where c_t represents the calcium concentration signal, b_t is a slowly varying baseline accounting for background auto-fluorescence and scattering, ϵ_t is measurement noise (due to acquisition noise, or imprecision in the localization and extraction of the signal), and a, b are scaling and offset parameters.

In addition, the calcium signal c_t is modeled as an auto-regressive model of order p with a sparse additive spiking signal s_t (typically modeling an exponential decrease with $p = 1$ and $\alpha_1 < 1$) [31, 32]:

$$c_t = \sum_{i=1}^{p} \alpha_i c_{t-i} + s_t \tag{I.2}$$

The objective is to recover a sparse, positive spike signal s_t that best fit the measured signal y_t. To achieve this, detrending and denoising are commonly applied to remove baseline drift and high-frequency noise, using a combination of low-pass and high-pass filters [22, 33]. Next, the spike inference problem is solved via deconvolution with L1-regularization on s_t to promote sparsity [31,32]. In TRASE-IN, we relied on the constrained Foopsi algorithm [31] implemented in CaImAn library [34].

Due to imaging noise and potential signal contamination from nearby neurons (see Figure I.6b), the deconvolution may yield spurious low-amplitude spikes. To address this, adaptive thresholding is applied to retain only significant spikes, resulting in a sparse, binary spike signal for each neuron.

The complete spike extraction pipeline implemented in TRASE-IN is illustrated in Figure I.11. The final output, referred to as *raster plot*, is a sparse binary matrix of size $n \times T$, where n is the number of neurons and T is the number of frames, indicating the inferred spiking times for each neuron.

I.2.4.3 Ensemble Identification

In Hydra, neurons tend to fire synchronously in groups known as functional ensembles [10]. Such synchronous spiking patterns are evident in the raster plot of Figure I.11.F, where approximately half of the recorded neurons participate in the CB (contraction burst) ensemble, firing together at frames 249, 421, 572, 712, and 808 [22]. Identifying these ensembles is a crucial step in understanding the neuronal coding underlying Hydra's behaviors, as it reduces the complexity of interpreting hundreds of individual neuronal activities to a few (≤ 5) collective functional units.

Let $s^i_{1:T} \in \{0,1\}^T$ denote the binary spiking vector for neuron i over T frames. A key initial step toward identifying ensembles is to define a pairwise distance or coupling measure between neurons that reflects their tendency to fire together. Due to small inaccuracies and temporal jitter in spike extraction, direct computation of correlations on the raw binary vectors $s^i_{1:T}$ often fails to capture true synchrony. To address this, spike trains are commonly convolved with Gaussian kernels prior to correlation analysis, allowing for temporal flexibility in co-activation detection [35, 36]. Alternatively, advanced statistical methods that explicitly account for spike timing offsets, such as SODA, can be employed for robust coupling estimation [37].

Once a coupling matrix has been computed, dimensionality reduction techniques such as Principal Component Analysis (PCA) can be applied to identify dominant ensembles [36,38,39]. Graph-based community detection methods, including the Louvain algorithm [37, 40], provide an alternative by clustering neurons based on their co-activation graphs. More sophisticated approaches leverage Bayesian inference frameworks incorporating biological priors to infer ensemble structures directly from spiking activity [23,41]. For a comprehensive review of statistical methods for ensemble identification in neuronal data, we refer the reader to the work of Kubler et al. [23].

I.2.4.4 Behavioral Correlation

The ultimate goal of this pipeline is to identify patterns in neuronal activity that underlie and trigger specific behaviors. This requires recording and classifying the animal's behavior and correlating these behavioral events with the ensemble activity patterns extracted from the neuronal data.

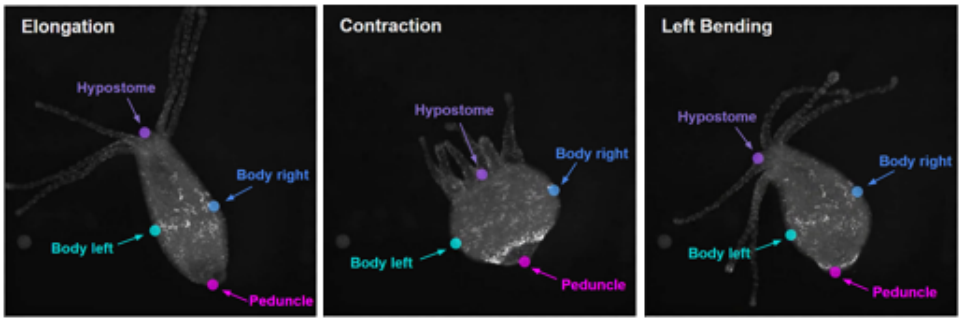


Figure I.12: Key-point tracking in Hydra using DeepLabCut [42], from the unpublished work of Evanne Martin, supervised during this PhD project. Key points on the head, mid-body, and foot are tracked to facilitate automatic behavioral classification.

Behavior Extraction While behavioral classification can be performed manually, recent deep learning-based pose estimation methods such as DeepLabCut [42], LEAP [43], or SLEAP [44] now allow for fast, accurate, and scalable tracking of animal postures, enabling automated behavior classification [45]. Early in this PhD project, we supervised an intern, Evanne Martin, to track key anatomical landmarks (head, foot, mid-body) of Hydra using DeepLabCut (Figure I.12). By applying an ad-hoc classifier on distances and angles between these tracked points, we successfully classified each video frame into distinct behaviors such as contraction, elongation, and bending.

Behavioral Correlation Once behavioral states are assigned to each frame, they can be directly correlated with the ensemble activity extracted earlier.

A straightforward analysis involves assessing the co-occurrence between specific neuronal ensemble activations and behaviors. For example, [45] demonstrated that RP1 ensemble neurons actively participate in Hydra somersaulting, with RP1 activity increasing before motion initiation and CB ensemble activation occurring prior to the reattachment of the foot (Figure I.13). In Zebrafish [23, 46] and mice [47], optical stimuli are applied to the animal with different grading/angles/amplitude. The different neuronal ensembles are shown to activate for specific optical stimuli.

More advanced approaches aim to predict behaviors directly from neuronal activity patterns, therefore demonstrating the encoding of behavioral information within the neural signals. For instance, in [48], long short-term memory (LSTM) networks [49] have been used to infer an animal's position from its neuronal activity.

At the beginning of this PhD project, Evanne Martin successfully trained an LSTM model on the global neuronal fluorescence signal (averaged across all neurons, as reliable individual tracking was not yet established) and demonstrated its ability to predict contraction behaviors in Hydra from collective neuronal activity patterns.

I.3 Goals

Robust tracking of thousands of neurons is a critical first step toward the end-to-end analysis of neuronal activity in freely behaving *Hydra vulgaris*. This PhD project was motivated by the limitations of classical multiple object tracking (MOT) algorithms (see Chapter II.4), which struggles to maintain reliable tracks over long sequences (typically >100 frames). These methods often produces identity switches—where a single track follows different neurons—or

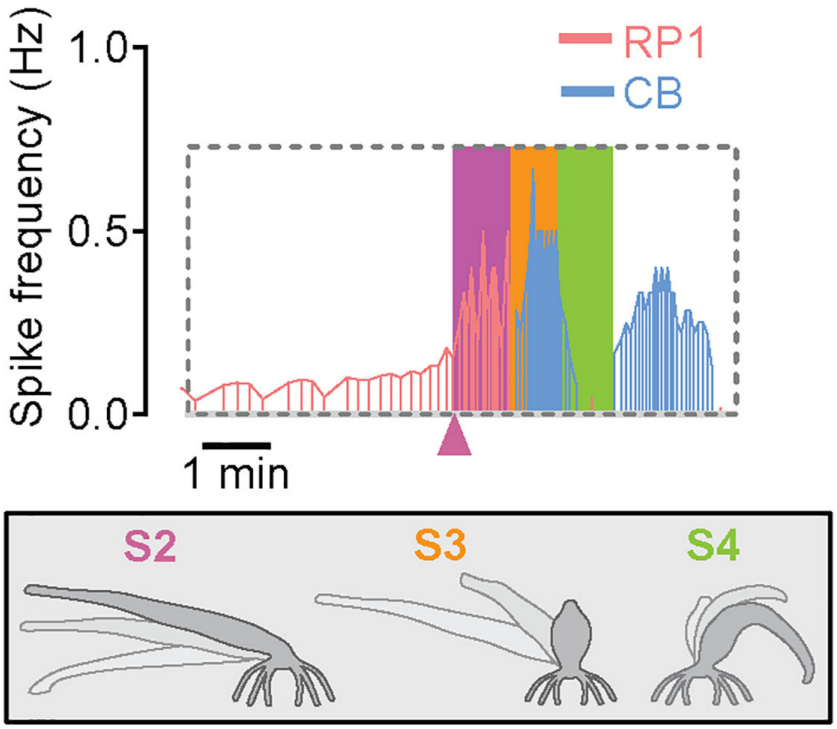


Figure I.13: Somersault behavior vs neuronal activation from [45]. The spike frequency of RP1 (red) and CB (blue) neuronal ensembles is shown alongside a somersault (S2, S3, S4) in Hydra. The data suggest that somersaults are associated with specific activation patterns: RP1 activity ramps up before movement initiation, followed by CB ensemble activation prior to foot reattachment.

fragmented trajectories. Such errors severely limit downstream analyses, which rely on accurate and continuous neuronal tracking.

The primary objective of this PhD was thus to develop more robust tracking methods tailored to the specific challenges of *Hydra vulgaris* neuronal recordings. While the Hydra use case motivated our work, the methods developed in this thesis were designed to generalize to other biological targets, such as cells or molecules. To improve adaptability and generalization across domains, we focused on unsupervised or self-supervised approaches, which do not rely on annotated training data, that are often scarcely available in biology.

We also placed a strong emphasis on usability, particularly for researchers without extensive programming backgrounds. As a result, we released clean, open-source, and well-documented implementations of our algorithms through the Python library `ByoTrack`[3], which includes reusable scripts and Jupyter notebooks. Integration of this library into popular image analysis platforms such as Icy [50], Fiji [51], and Napari [52] has also been initiated.

I.4 Challenges and Contributions

In this section, we present the major challenges associated with MOT in the context of Hydra neuron tracking and summarize the contributions of this thesis that address them.

[3] `https://github.com/raphaelreme/byotrack`

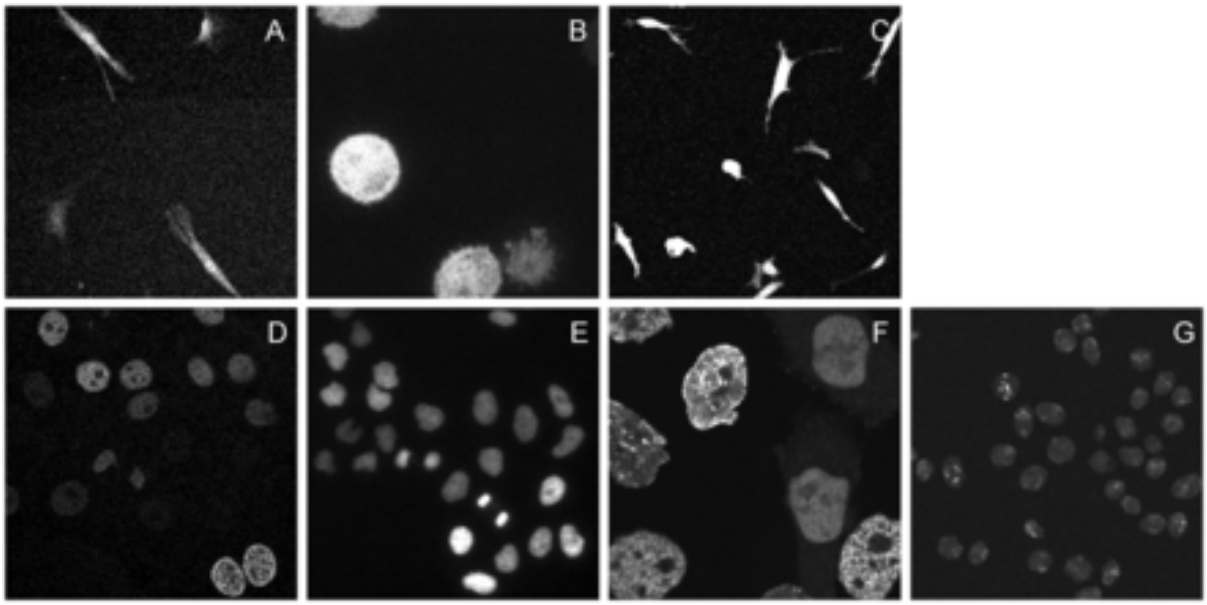


Figure I.14: Cell Tracking Challenge (CTC) datasets from [59]. Representative regions from the video dataset repository. **(A)** C2DL-MSC; **(B)** C3DH-H157 (selected z-slice); **(C)** C3DL-MDA231 (selected z-slice); **(D)** N2DH-GOWT1; **(E)** N2DL-HeLa; **(F)** N3DH-CHO (selected z-slice); **(G)** N2DH-SIM (also representative of a selected z-slice of N3DH-SIM)

I.4.1 Lack of Annotated Tracking Data

Annotated tracking datasets in biology are rare and often specific to narrow use cases. The labeling process is time-consuming and prone to human errors and biases, especially for tracking, which requires long-term annotations across multiple objects and frames [53, 54]. In contrast, tasks such as image classification or segmentation can be evaluated on a few regions of interest in a small number of images. Tracking, by nature, demands at least spatial and temporal consistency across a few several sequences. Consequently, many biological tracking methods are evaluated quantitatively on synthetic datasets and only qualitatively on experimental data, which often lacks ground truth annotations [27, 53–58].

I.4.1.1 Data for Supervised Learning

With the recent advances in deep learning, supervised learning has become the standard in MOT for natural images (e.g., TrackFormer [60]). Howerever, its use in biological applications remains limited due to the scarcity of annotated data and the lack of distinctive features (e.g., texture, color) in targets such as neurons, cells, pathogens or molecules. The Cell Tracking Challenge (CTC) [59, 61] remains the largest publicly available dataset for biological tracking (Figure I.14), comprising ∼40 videos with public annotations. In this benchmark, deep learning methods have shown strong performance for detection and segmentation and have also been applied in tracking for motion modeling through methods like Linajea [62], EmbedTrack [63], and TrackAstra [64]. However, generalization across datasets remains limited and mostly applies to cells undergoing Brownian motion and with cellular division, conditions quite different from neuron tracking in Hydra.

Given the limited applicability of supervised methods in this context, all methods developed during this PhD were either unsupervised or self-supervised (i.e., trained using pretext tasks without ground truth, allowing them to be retrained freely on new data).

I.4.1.2 Data for Quantitative Evaluation

Even unsupervised or self-supervised methods require annotated data for quantitative evaluation. Unfortunately, no such dataset exists for neuron tracking in Hydra. Existing resources, such as the CTC [61] or the Particle Tracking Challenge (PTC) [54], focus on synthetic particles or dividing cells *in vitro* with independent motion patterns. They do not reflect the global, sudden and elastic motion of neurons in a moving organisms.

To address this, we developed our own synthetic dataset, SINETRA [65] (Chapter III), which simulates 2D and 3D time-lapse sequences mimicking the complex motion of neurons in behaving animals. This allows for reproducible benchmarking and objective evaluation of tracking algorithms tailored to Hydra.

I.4.2 Tracking Challenges in Hydra

In this section, we summarize the major challenges of tracking neurons in Hydra, ranging from general issues in biological imaging to specific properties of this organism. Some of these challenges are illustrated in Chapter II.4.

I.4.2.1 Signal-to-Noise Ratio

As with many biological datasets, GCaMP and TdTomato images of Hydra are acquired via fluorescence microscopy. Despite significant progress in imaging techniques, these data are still limited by low signal-to-noise ratio (SNR). This is particularly challenging in behaving animals, where motion blur and phototoxicity constrain the imaging frame rate, which must balance temporal resolution against noise levels.

In particular, neurons in GCaMP images (Figure I.7) can be difficult to distinguish from background noise—even for expert annotators. Although the TdTomato channel improves neuron visibility (Figure I.8), choosing robust segmentation thresholds remains challenging. A low SNR reduces the ability of both human annotators and automatic algorithms to accurately segment, track, or differentiate between individual neurons.

I.4.2.2 Neuron Density

Some neurons are densely packed, especially in regions such as the animal's foot or head (Figure I.8). In such cases, threshold-based segmentation approaches frequently merge adjacent neurons, resulting in under-segmentation (see Chapter II.1.2.1 and Figure I.10). Although deep learning methods such as StarDist [25] can mitigate this issue, they require segmentation annotations.

Moreover, a successful tracking requires accurate motion prediction to maintain identity consistency across frames. This becomes especially difficult when neurons are close together and exhibit unexpected motion.

In this thesis, we did not try to improve the segmentation methods and rather rely on existing segmentation algorithms, such as the wavelet spot detector [24], or StarDist where applicable (*i.e.* with a suitable training dataset). Instead, we designed robust tracking-by-detection algorithms, able to track despite detections errors.

I.4.2.3 Animal Motion

Hydra exhibits rapid and large-scale deformations, especially in the tentacles and head during contraction (Figure I.3). Standard motion models, such as Kalman filters, often fail to predict such sudden motion accurately, leading to incorrect track associations and identity switches.

To address this, we propose a new motion model in Chapter IV that incorporates optical flow as a second, image-based velocity sensor to enhance Kalman filter predictions [66].

I.4.2.4 Intermittent Visibility

Due to the activity-dependent nature of GCaMP fluorescence, inactive neurons intensity frequently drops below detection thresholds and such neuron may remain undetected for over 100 consecutive frames. In these cases, standard MOT methods tend to produce fragmented short tracks referred to as *tracklets*. These tracklets must be linked into full-length trajectories using a dedicated *tracklet stitching* post-processing step.

In previous work, Thibault Lagache proposed a geometric stitching method to link such tracklets across elastic deformations [28]. We extended this method by incorporating self-supervised visual features to improve robustness, as detailed in Chapter VI.

I.4.3 Contributions & Publications

This PhD thesis presents several contributions toward the robust tracking of neurons in *Hydra vulgaris*, enabling fine-grained (per-neuron) recording and analysis of neuronal activity in correlation with the animal's behavior.

At the beginning of this project, we established an analysis pipeline for dual-labeled Hydra (expressing both GCaMP and TdTomato), which we named TRASE-IN (Chapter I.2.4). In parallel, we developed methods to improve tracklet stitching using self-supervised deep learning to learn neuron-specific visual features (Chapter VI). These two initial efforts revealed limitations in the existing MOT algorithms, particularly in their ability to cope with the organism's large-scale deformations. To address this, we developed SINETRA (Chapter III), a synthetic dataset that reproduces realistic motion and noise challenges, allowing for objective comparison of neuron tracking algorithms in behaving animals. Then, we introduced KOFT (Chapter IV), a novel tracking method that incorporates optical flow as a second independent velocity sensor into a Kalman filtering framework.

Finally, we extended our KOFT tracking framework into a fully Bayesian Multiple Hypothesis Tracking (MHT) framework. We tailored our Bayesian framework to rely on few user-defined parameters, that can be manually tuned without expect knowledge. This Bayesian KOFT (KOFT-MHT) strongly improves tracking robustness for our Hydra use case (Chapter V).

Throughout this work, we rely on existing biological object detection methods. Our default approach is a wavelet-based enhancement and thresholding technique [24] (see Chapter II.1.2.1). We also trained deep learning-based detectors (e.g., StarDist) on TdTomato-labeled images. Although tracking performance is highly dependent on the quality of detection, we deliberately focused our contributions on the data association and linking task, which cannot be tackled through supervised learning due to limited annotated data and, on which classical MOT approaches struggle due to the specific challenges listed above.

All tools developed during this thesis are open source, and the majority are distributed

via the Python package `ByoTrack`[4], which includes reusable code, notebooks and scripts for detection and tracking in both 2D and 3D. We also extended this library to support cell division events and submitted our tracking algorithms to the Cell Linking Benchmark (CLB)[5] of the Cell Tracking Challenge (CTC), where one of our unsupervised method, initially designed for Hydra's neurons, ranked first among seven competitors, outperforming supervised approaches on these datasets. This result highlights the generalization potential of our unsupervised methods for biological MOT.

I.4.3.1 Publications

Below is the complete list of publications resulting from this thesis work, ordered chronologically. (Co-)First authors are denoted with a *.

- **Visual Tracklet Stitching (Chapter VI)**: Raphael Reme*, Victor Piriou, Alison Hanson, Rafael Yuste, Alasdair Newson, Elsa Angelini, Jean-Christophe Olivo-Marin, and Thibault Lagache.
  Tracking intermittent particles with self-learned visual features. IEEE ISBI, 2023. [67]
- **TRASE-IN (Chapter I.2.4)**: Alison Hanson*, Raphael Reme*, Noah Telerman*, Wataru Yamamoto, Jean-Christophe Olivo-Marin, Thibault Lagache, and Rafael Yuste.
  Automatic monitoring of neural activity with single-cell resolution in behaving Hydra. Scientific Reports, 2024. [22]
- **KOFT (Chapter IV)**: Raphael Reme*, Alasdair Newson, Elsa Angelini, Jean-Christophe Olivo-Marin, and Thibault Lagache.
  Particle tracking in biological images with optical-flow enhanced Kalman filtering. IEEE ISBI, 2024. [66]
- **SINETRA (Chapter III)**: Raphael Reme*, Alasdair Newson, Elsa Angelini, Jean-Christophe Olivo-Marin, and Thibault Lagache.
  SINETRA: A Versatile Framework for Evaluating Single Neuron Tracking in Behaving Animals. IEEE ISBI, 2025. [65]
- [SUBMITTED] **KOFT Journal Extension (Chapter IV)**: Raphael Reme*, Alasdair Newson, Elsa Angelini, Jean-Christophe Olivo-Marin, and Thibault Lagache.
  Using Optical Flow as a Velocity Sensor for Kalman-Based Tracking of Biological Objects. Submitted to IEEE TIP.
- [IN PREPARATION] **KOFT-MHT (Chapter V)**: Raphael Reme*, Alasdair Newson, Elsa Angelini, Jean-Christophe Olivo-Marin, and Thibault Lagache.
  Bayesian Multiple Hypothesis Tracking with Optical flow enhanced Kalman filtering To be submitted to IEEE ISBI 2026.

I.4.3.2 Manuscript Outline

This manuscript is structured as follows.

In **Chapter I - Introduction**, we present the general context of the PhD, outlining its goals and motivation within the broader project of neuronal activity analysis in behaving animals like *Hydra vulgaris*. We rely partially on our worked published in Scientific Reports:

[4] `https://github.com/raphaelreme/byotrack`
[5] `https://celltrackingchallenge.net/latest-clb-results/`

- Alison Hanson*, Raphael Reme* et al. *Automatic monitoring of neural activity with single-cell resolution in behaving Hydra.* Scientific Reports, 2024. [22]

Chapter II - Multiple Object Tracking extensively reviews the state of the art in MOT, with a particular focus on biological targets. We conclude with a detailed example of neuron tracking in Hydra to highlight the challenges addressed in this thesis.

Chapter III - SINETRA introduces our synthetic dataset for evaluating MOT applied to neurons in behaving animals. We benchmark multiple algorithms and discuss how SINETRA enables fair and reproducible comparison. This work was published in ISBI 2025:

- Raphael Reme* et al., *SINETRA: A Versatile Framework for Evaluating Single Neuron Tracking in Behaving Animals*, IEEE ISBI 2025. [65]

In **Chapter IV - KOFT**, we describe a novel method that incorporates optical flow as a second sensor in Kalman filtering to improve motion prediction. This work was presented at ISBI 2024 and submitted to IEEE TIP:

- Raphael Reme* et al., *Particle tracking in biological images with optical-flow enhanced kalman filtering*, IEEE ISBI 2024. [66]
- Raphael Reme* et al., *Using Optical Flow as a Velocity Sensor for Kalman-Based Tracking of Biological Objects*, Submitted to IEEE Transactions on Image Processing

Chapter V - KOFT-MHT extends our tracking framework in a fully Bayesian solver (MHT) to improve robustness in densely packed and ambiguous tracking scenarios. To be submitted to ISBI 2026.

Chapter VI - VEMC2, proposes enhancements to the EMC2 tracklet stitching method, using visual features learned via self-supervised learning. This work was published in ISBI 2023:

- Raphael Reme* et al., *Tracking intermittent particles with self-learned visual features*, IEEE ISBI 2023. [67]

Finally, **Chapter VII - Conclusion and Perspectives** summarizes the thesis contributions, discusses limitations, and highlights potential future directions, including broader applications of the developed tracking algorithms.

Chapter II

Multiple Object Tracking: State of the Art

Multiple Object Tracking (MOT) aims to track individual objects in 2D or 3D videos. The objects of interest, or "targets", are assumed to be visible in at least some frames of the video, making it possible to estimate their pixel-wise positions over time.

The input to an MOT algorithm is a video of length T, represented as a sequence of T images: $I_{1:T} = (I_t)_{1 \leq t \leq T}$. Each image I_t is defined as a mapping from a spatial coordinate (pixel) $\mathbf{w} \in \mathbb{R}^d$ to an intensity value $I_t(\mathbf{w}) \in \mathbb{R}^c$, where d is the spatial dimension of the image (e.g., $d = 2$ for standard images), and c is the number of channels (e.g., $c = 3$ for RGB images).

In practice, images are defined only on a discrete, bounded domain. Let $S \in \mathbb{N}^d$ denote the size of the image along each dimension. The image is then truly defined only on the discrete pixel grid $\mathbb{W} = \bigotimes_{k=1}^{d} [\![0, S_k - 1]\!]$ (a Cartesian product of intervals), such that $\mathbb{W} \subset \mathbb{N}^d \subset \mathbb{R}^d$. Values at other locations can be obtained via interpolation or extrapolation. For simplicity, in all following of the thesis, we extend the definition of an image as a mapping from $\mathbb{R}^d$ to $\mathbb{R}^c$.

Targets are assumed to be *visible* in at least one frame, meaning their presence alters the values of a subset of pixels, thereby allowing for their localization. The objective of an MOT algorithm is to determine the location of all targets across all frames of the video.

For the sake of generality, we assume that the i^{th} target at time t is described by a state vector $\mathbf{x}_t^i \in \mathbb{R}^{d_\mathbf{x}}$, where $d_\mathbf{x}$ is the dimension of the state space. This state encodes all relevant information needed to model the target's appearance in the image, such as position, velocity, size, color, and other visual features. In practice, the definition of the state varies across methods, from a simple 2D position (e.g., the target's centroid) to more complex representations like parametric curves, segmentation masks, or visual descriptors.

Let N be the number of targets in the video, which is typically unknown and must be estimated. Let $X_t = (\mathbf{x}_t^i)_{1 \leq i \leq N}$ be the state of all targets at time t and $X_{1:T} = (X_t)_{1 \leq t \leq T}$ be the full sequence of states over time. MOT algorithms aim to estimate the sequence $X_{1:T}$ that best *explains* the observed video data, where the definition of "best" depends on the specific algorithmic approach.

In this framework, a *track* can be defined as the sequence of states of a single target across time: $X_{1:T}^i = (\mathbf{x}_t^i)_{1 \leq t \leq T}$.

In the following sections, we review various approaches to address this task, with a particular

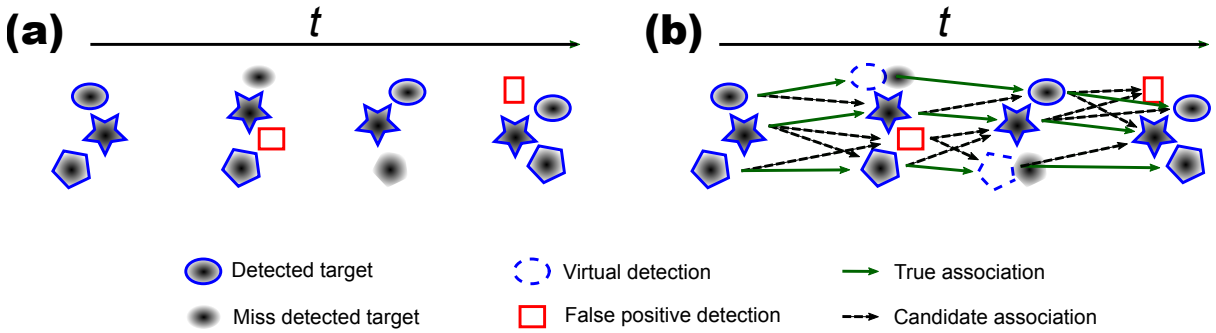


Figure II.1: Illustration of the tracking-by-detection paradigm with three targets (circle, star, and pentagon) across four frames ($t = 1$ to $t = 4$). **(a)** Detections are shown for each frame. The circle target is missed on frame 2, and the pentagon target on frame 3. False positive detections (in red) appear on frames 2 and 4. **(b)** Linking detections to reconstruct tracks under the 1-to-1 association assumption. Tracks are formed by selecting consistent paths (green edges) from a set of candidate associations. The linking algorithm must bridge missed detections—e.g., by introducing *virtual detections*—and reject false positives from being included in the final tracks.

focus on tracking-by-detection strategies and Bayesian methods, which form the basis of the algorithms developed in this thesis.

II.1 Tracking-By-Detection framework

As multiple object tracking (MOT) is a particularly challenging task, a widely adopted strategy is to decompose it into two simpler sub-tasks: (i) the detection of targets in each individual frame, followed by (ii) the association (or linking) of these detections across time. This paradigm, known as *tracking-by-detection* benefits from recent advances in object detection and segmentation—particularly with the advent of deep learning. Furthermore, the linking step can be formulated as an optimization problem that naturally accommodates the appearance and disappearance of targets, therefore enabling estimation of the (often unknown) number of targets N in the video.

The detections process is prone to errors. In particular, some targets are miss detected because of imaging conditions (e.g. low SNR, particles moving out-of-focus, occlusions...) or the biophysics of the imaged target, that can switch into a non-detectable state (e.g. intermittency of Quantum Dots fluorescence [68], photo-switchable fluorophores [69], calcium imaging of neurons [70]...). Also noise and background structures are often detected as targets, giving false positive detections, also referred to as *clutter* detections. Figure II.1 shows a simple example of such cases, where the linking algorithm have to handle and correct for detections errors.

In this thesis, we adopt the tracking-by-detection framework and in particular, we propose novel linking algorithms detailed in Chapters IV and V. In this section, we first introduce the formalism of this approach, then review standard detection techniques, and finally present various strategies and algorithms for linking.

II.1.1 Formalism

II.1.1.1 Detection

A detection algorithm extracts a set of measurements or detections $Z_t = (\mathbf{z}_t^j)_{1 \leq j \leq M_t}$ for each input frame I_t, where M_t denotes the number of detections at time t and $\mathbf{z}_t^j$ is the j^{th} measurement

vector. The sequence of detections sets across time is denoted by $Z_{1:T} = (Z_t)_{1 \leq t \leq T}$.

Each detection vector $\mathbf{z}_t^j \in \mathbb{R}^{d_\mathbf{z}}$ lies in a measurement space of dimension $d_\mathbf{z}$ and encodes the localization of the detected target. We distinguish three primary detection formats:

1. **Center of mass:** Each detection is represented by the coordinates of its center of mass. In this case, $d_\mathbf{z} = d$, where d is the spatial dimensionality of the image. For instance, in 2D we have $\mathbf{z}_t^j = (x, y) \in \mathbb{R}^2$. This representation is particularly suited for point-like targets (referred to as *particles*), and can also be used for larger targets for simplicity.

2. **Bounding box:** Each detection is encoded as a d-dimensional bounding box, typically specified by a reference point (e.g., center or top-left corner) and its spatial dimensions. This yields $d_\mathbf{z} = 2d$, e.g., in 3D: $\mathbf{z}_t^j = (x, y, z, dx, dy, dz) \in \mathbb{R}^6$. This format provides the spatial extent of detected targets while maintaining a compact representation.

3. **Segmentation mask:** Each detection is a binary mask indicating, for every pixel $\mathbf{w} \in \mathbb{W}$, whether it belongs to the detected object. Formally, $\mathbf{z}_t^j \in [\![0,\, 1]\!]^{\mathbb{W}}$. While this format offers a rich, fine-grained description of targets localization, it is also much more complex and is often converted to simpler representations for the linking.

Although detection algorithms may use a specific internal format, it can typically be transformed to the one required by the linking algorithm. Moreover, the measurement vector can be augmented with handcrafted (e.g. intensity, size, curvature, ...) or learnt (e.g. with deep learning) features prior to the linking phase.

II.1.1.2 Linking

The linking algorithm connects detections across time to form coherent trajectories. Each detection in Z_t is indexed from 1 to M_t. Let $J_t = [\![1,\, M_t]\!]$ be the set of detection indices at time t and $J_{1:T} = (J_t)_{1 \leq t \leq T}$ be the full temporal sequence of detections indices. A track for target i is characterized by a sequence of index sets $\gamma_{1:T}^i = (\gamma_t^i)_{1 \leq t \leq T}$, with:

$$\forall t \in [\![1,\, T]\!]\,, \gamma_t^i \subset J_t \tag{II.1}$$

where γ_t^i is the subset of detection indices associated with target i at time t.

This very general formalism allows to handle the errors from the detection process. Indeed for non-existing or miss-detected targets, then no detection can be assigned to the track with $\gamma_t^i = \emptyset$. In cases of a split detection, multiple detections may correspond to the same target ($|\gamma_t^i| > 1$). Conversely, a single detection may be associated with multiple targets (merged detection), *i.e.*, $\exists i_1 \neq i_2$ such that $\gamma_t^{i_1} \cap \gamma_t^{i_2} \neq \emptyset$. Finally, false positive detections are simply not associated with any track.

For convenience, we refer to the sequence $\gamma_{1:T}^i$ as a "track". The set of all feasible tracks given the detections indices $J_{1:T}$ is denoted $\Omega(T)$, which can be written as a cartesian product (where $\mathsf{P}(S)$ denotes the power set of S):

$$\Omega(T) = \bigotimes_{t=1}^{T} \mathsf{P}(J_t) \tag{II.2}$$

For each track $\gamma_t^i \in \Omega(T)$, its *birth* and *death* times are defined as its first and last appearance times (see tracks 2 and 3 from Figure II.2):

$$t_{\mathrm{b}}^i = \min\{t \mid |\gamma_t^i| \geq 1\} \tag{II.3}$$

$$t_{\mathrm{d}}^i = \max\{t \mid |\gamma_t^i| \geq 1\} \tag{II.4}$$

The set of detection vectors associated with track i at time t is denoted:

$$Z_t^{\gamma_t^i} = \{\mathbf{z}_t^j \mid j \in \gamma_t^i\} \tag{II.5}$$

and the full sequence across time is $Z^{\gamma_{1:T}^i} = (Z_t^{\gamma_t^i})_{1 \leq t \leq T}$.

The linking stage outputs a subset of N tracks over T frames, denoted $\Gamma_{1:T} = (\gamma_{1:T}^i)_{1 \leq i \leq N}$. Given a subset of tracks $\Gamma_{1:T} \subset \Omega$, we define the associated detections indices at time t as $J_t^{\mathrm{a}} = \bigcup_{i=1}^N \gamma_t^i \subset J_t$, and the set of unassigned detections indices (i.e., false positives or *clutter*) as $J_t^{\mathrm{f}} = J_t \setminus J_t^{\mathrm{a}}$. The sequence of false positive detections is then denoted $J_{1:T}^{\mathrm{f}} = (J_t^{\mathrm{f}})_{1 \leq t \leq T}$. For instance after the linking, false positive detections are colored in red in Figures II.1.b and II.2.d.

A widely used and equivalent formulation of the linking is the measurement-oriented one, where we seek to assign labels $L_t = (L_t^j)_{1 \leq j \leq M_t}$ to each detection, with:

$$L_t^j = \{i \in [\![1, N]\!] \mid j \in \gamma_t^i\} \tag{II.6}$$

A detection is labeled with the indices of tracks it is associated with. False positives satisfy $L_t^j = \emptyset$, while merged detections can have $|L_t^j| > 1$ (see Figure II.2).

This measurement-oriented formulation is fully equivalent to the track-oriented one, and both may be used interchangeably. Specifically, one can recover the track associations from labels via $\gamma_t^i = \{j \mid i \in L_t^j\}$.

A widely adopted simplification in many tracking algorithms is the *1-to-1 association* assumption. Under this assumption, each detection is associated with at most one target, and each target is linked to at most one detection per frame:

$$|L_t^j| \leq 1, \quad |\gamma_t^i| \leq 1 \tag{II.7}$$

This constraint removes the possibility of modeling merged or split detections. When a detection corresponds to multiple true targets (e.g., overlapping cells in a single bounding box), only one target may be associated with it—the others are treated as missed. Conversely, when a single target generates multiple detections (e.g., an over-segmented cell), only one detection is retained, and the others are classified as false positives.

Linking algorithms solve an optimization problem of the form:

$$\Gamma_{1:T}^\star \in \underset{\Gamma_{1:T} \subset \Omega(T)}{\arg\min}\ \mathcal{C}(\Gamma_{1:T}, Z_{1:T}) \tag{II.8}$$

where $\mathcal{C}$ is a cost function that quantifies how well the proposed associations $\Gamma_{1:T}$ fit the observed detections $Z_{1:T}$. Typically, the cost is low when tracks are associated to detections that share similar features. When tracking natural image targets (human, cars, planes, ...) visual features (like clothes, colors, brand, ...) can be extracted to discriminate between objects. In biology, these are less discriminative, and the cost is rather based on shape, intensity and mostly positional features (low, or smooth target motion).

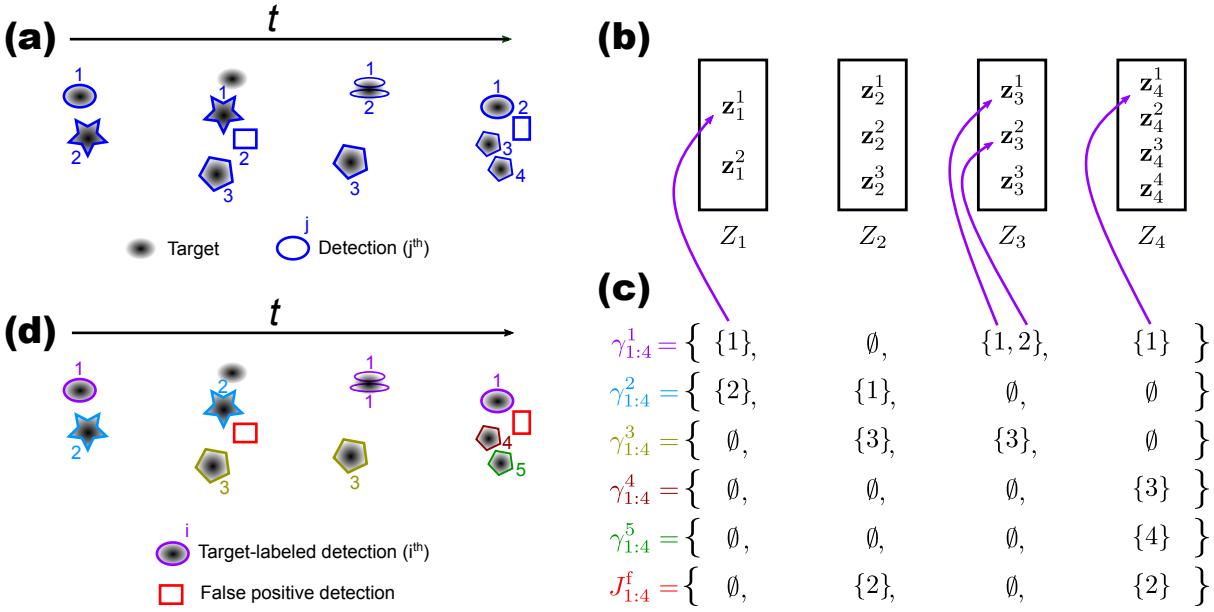


Figure II.2: Illustration of track-oriented and measurement-oriented linking formalism, highlighting key challenges in object tracking: false positives, missed detections, and target/detection splitting. **(a)** Detections for each of the four frames are shown, with arbitrary indexing from top to bottom. The circle target is missed on frame 2, and false positive detections (shown as rectangles) appear on frames 2 and 4. On frame 3, two detections (1 and 2) correspond to the same circle target (i.e., a split detection). **(b)** Detections sets $Z_{1:4}$ corresponding to the top-left panel. **(c)** True labeling of detections using track-oriented formalism. Each track is associated with unique color (see on bottom left panel), while red denotes the set of false positives. For instance, the track 1 (circle, purple) starts on frame 1 with the detection first detection $\mathbf{z}_1^1$ ($\gamma_1^1 = \{1\}$), it is miss-detected on frame 2 ($\gamma_2^1 = \emptyset$), then associated to the two first detections of frame 3 ($\gamma_3^1 = \{1, 2\}$) and finally associated to $\mathbf{z}_4^1$ ($\gamma_4^1 = \{1\}$). The matching between the first track indices and detections is illustrated by purple arrows. A birth event occurs for the pentagon target on frame $t_{\mathrm{b}}^3 = 2$, and a death event for the star target on frame $t_{\mathrm{d}}^2 = 2$. On frame 4, the pentagon undergoes a target splitting event, producing two new targets (4 and 5). **(d)** True labeling of detections using measurement-oriented notation. Detections are shown with their associated track identifier. Again, each color represents a unique target, while red denotes false positives: *e.g.*, detection 2 on frames 2 and 4 are unassigned: $L_2^2 = L_4^2 = \emptyset$. On frame 3 the two first detections are associated to track 1 (circle, purple) and the last detection is associated to track 3 (pentagon, yellow): $L_3^1 = \{1\}$, $L_3^2 = \{1\}$, $L_3^3 = \{3\}$.

In essence, a linking algorithm is defined by two components: a cost function $\mathcal{C}$ and a strategy to minimize it. We note that without further assumptions and heuristics, the cardinality of Ω grows exponentially in time and number of targets, preventing to find $\Gamma^{\star}_{1:T}$ efficiently.

A common structure of the cost function is the following decomposition:

$$\mathcal{C}(\Gamma_{1:T}, Z_{1:T}) = \sum_{i=1}^{N} C(Z^{\gamma^i_{1:T}}) + \sum_{t=1}^{T} \sum_{j \in J^{\mathrm{f}}_t} c^{\mathrm{f}}(\mathbf{z}^j_t) \tag{II.9}$$

$$= \sum_{i=1}^{N} \sum_{t=1}^{T} c(Z^{\gamma^i_t} \mid Z^{\gamma^i_{1:t-1}}) + \sum_{t=1}^{T} \sum_{j \in J^{\mathrm{f}}_t} c^{\mathrm{f}}(\mathbf{z}^j_t) \tag{II.10}$$

where $C(Z^{\gamma^i_{1:T}})$ denotes the total cost of a single track i, which is often decomposed into per-frame propagation costs $c(Z^{\gamma^i_t} \mid Z^{\gamma^i_{1:t-1}})$. The second term penalizes false positive detections via a cost c^{f} for each unassociated detection.

Note that before a track begins ($t < t^i_{\mathrm{b}}$) and after it ends ($t > t^i_{\mathrm{d}}$), the propagation cost is assumed to be zero:

$$t \notin [\![t^i_{\mathrm{b}}, t^i_{\mathrm{d}}]\!] \Longrightarrow c(Z^{\gamma^i_t} \mid Z^{\gamma^i_{1:t-1}}) = 0 \tag{II.11}$$

Special care is often taken in modeling the costs at the birth and death of targets (see track 2 and 3 from Figure II.2), which are relatively rare events and may incur higher penalties. Beyond simple birth/death events, some scenarios require modeling more complex interactions between tracks, such as:

- **Target splitting:** A single target at time t may give rise to multiple targets at time $t+1$. This occurs, for example, in biological imaging when a cell divides (mitosis). The model must allow for such events and assign appropriate costs to encourage or discourage them, depending on context. An example of target splitting is given in Figure II.2, where track 3 splits into tracks 4 and 5.
- **Target merging:** Conversely, multiple targets at time t may merge into a single target at time $t+1$. This happens in scenarios such as receptor aggregation on cell membranes. Again, this can be modeled by specific cost terms.

Unlike merged or split detections—which do not alter the number of tracked targets—target splitting and merging events modify the number of active targets across time. Accurate modeling of these events is crucial in applications like biological imaging, where understanding such transitions is often of central importance.

II.1.1.3 State Estimation

After the association step has produced a consistent set of tracks $\Gamma_{1:T}$, the final objective is to estimate, for every target, a continuous state vector $\mathbf{x}^i_t \in \mathbb{R}^{d_{\mathbf{x}}}$ that summarizes its latent properties at each time instant (*e.g.*, sub-pixel position, velocity, orientation, size, or fluorescence intensity).

The most direct strategy is to derive the state from the raw measurements that belong to the track:

$$\mathbf{x}^i_t \;=\; g(Z^{\gamma^i_t}), \tag{II.12}$$

where $g(\cdot)$ is typically a component-wise average, median, or centroid operator and therefore $d_{\mathbf{x}} = d_{\mathbf{z}}$. If $\gamma_t^i = \emptyset$ (missed detection) the state can be flagged with a dummy vector $\mathbf{x}^{\varnothing}$ or interpolated from neighboring time frames.

Although Eq. (II.12) is adequate for high-SNR particle data, it neglects two crucial issues: (i) measurement noise corrupts every $\mathbf{z}$; (ii) physically meaningful states—such as velocity—cannot be obtained from a single frame. Both shortcomings are addressed by *stochastic state–space models*. This will be detailed in the Bayesian framework, where such state estimation is framed as a Bayesian smoothing problem (see Chapter II.3.2).

II.1.2 Detection Approaches

II.1.2.1 Signal Enhancement and Thresholding

One of the most widely used strategies for object detection in biomedical imaging consists of two stages: signal enhancement followed by thresholding. These methods aim to identify candidate objects in microscopy images, such as cells, nuclei, vesicles, or macromolecular complexes, by amplifying relevant structures and removing background noise.

Signal enhancement. Raw microscopy images often suffer from low signal-to-noise ratio, inhomogeneous background, or weak contrast. A variety of preprocessing filters can be used to enhance signal features of interest. The most common enhancement techniques include:

- **Gaussian filtering** for denoising and smoothing.
- **Top-hat filtering**, a morphological filter that highlights bright or dark spots relative to their surroundings.
- **Total Variation (TV-L1) denoising** [71, 72] to preserve edges while reducing noise.
- **Difference of Gaussians (DoG)** or **Laplacian of Gaussian (LoG)** filters to enhance blob-like structures at a given scale [73, 74].
- **Wavelet transforms** that enhance blob-like structures at multiple scales [24].

These enhancement steps yield a response map where high intensity regions correspond to potential object locations.

Thresholding. Once the signal is enhanced, a binary mask is created by applying a threshold to the response image. Threshold selection is critical and can be done:

- **Globally**, using user-defined values or automatic heuristics such as Otsu's method [75,76].
- **Locally**, using adaptive thresholds that vary across the image to handle illumination inhomogeneity.

After thresholding, the resulting binary mask identifies candidate object regions.

Post-processing with connected components. To convert the binary mask into a set of discrete detections, connected component labeling is applied. This step assigns a unique label to each spatially connected region, from which individual detection features (centroid, size, shape, intensity, etc.) can be extracted. In 2D, connectivity is typically defined using 4- or 8-connected neighborhoods; in 3D, 6-, 18-, or 26-connected components are common.

Optionally, additional steps such as size filtering, shape filtering, or watershed segmentation [77] can be used to refine the mask or separate overlapping objects.

The set of detections Z_t is simply the set of connected components, where each detection is represented by its segmentation mask.

Discussion. This two-step pipeline (enhancement + thresholding) is fast, interpretable, and easily tunable, making it widely applicable to microscopy images. However, it can be sensitive to parameter choices (e.g., kernel sizes, thresholds), image artifacts, and variability in object appearance. For large-scale datasets or challenging imaging conditions, more advanced detection approaches (e.g., learning-based methods) may offer improved robustness.

Still, signal-enhancement-based detection remains a strong baseline and is often used as a first step in many tracking-by-detection pipelines.

II.1.2.2 Model-Based

Model-based detection approaches use explicit geometrical or physical models to delineate object boundaries or estimate object parameters directly from the image data. Unlike threshold-based methods that rely on image intensity alone, model-based techniques incorporate prior knowledge about the expected shape, size, or topology of the objects to be detected.

Active contours and level sets

One of the most well-known model-based methods is the *active contour* or *snake* framework [78–82]. A snake is a deformable curve $\mathbf{C}(s)$ that evolves under internal forces (e.g., smoothness constraints) and external image-derived forces (e.g., gradients) to minimize an energy functional:

$$E[\mathbf{C}] = \int \left[\alpha|\mathbf{C}'(s)|^2 + \beta|\mathbf{C}''(s)|^2 + E_{\text{image}}(\mathbf{C}(s))\right] \mathrm{d}s \tag{II.13}$$

where E_{image} attracts the curve toward object boundaries, and α, β control elasticity and stiffness.

To handle topological changes (e.g., splitting or merging), *level set* methods [83–86] extend active contours by representing object boundaries implicitly as the zero level set of a higher-dimensional function $\phi(\mathbf{x})$. The evolution of ϕ is governed by partial differential equations that minimize a suitable energy functional (such as Mumford-Shah functional [87]), allowing the contour to adapt flexibly to complex shapes and topologies.

Initialization and optimization. A common challenge in snakes-based methods is the need for initialization. Since many of these methods optimize non-convex energy functions, their performance is highly dependent on a good initial guess. Initialization may be manual or derived from simple heuristics (*e.g.*, blob detection). The optimization process can also be computationally intensive, particularly in 3D or when applied to large numbers of objects.

Discussion. Snake-based detection methods offer fine-grained, accurate delineation of object boundaries and are especially well-suited to scenarios where object shape is regular or with clear boundaries. However, their dependence on initialization, sensitivity to noise, and computational complexity limit their applicability in large-scale biological applications. Moreover, many models do not extend well in 3D (*e.g.*, snakes) and requires dedicated 3D extensions [86].

These methods are often used as a post-processing to refine coarse object contours from simpler detection methods (*e.g.*, thresholding), enabling more precise shape analyses.

Parametric and generative models

Some approaches fit predefined parametric shapes (e.g., ellipses, spheres, rectangles) to candidate regions using optimization or probabilistic inference. These are particularly effective when the object shape is known a priori and approximately constant across instances (e.g., nuclei or vesicles). In more advanced settings, generative models may be used to simulate how the appearance of objects arises from underlying physical processes, and inverse inference is performed to estimate object parameters [88–92].

These methods provide precise segmentation of targets as long as a prior shape is known or can be estimated. Nonetheless, they usually require intensive optimization process that limits their usage for large scale datasets.

II.1.2.3 Deep Learning

Over the past decade, deep learning has emerged as the state-of-the-art paradigm for object detection, particularly in contexts where annotated training data is available. These approaches leverage large neural networks trained end-to-end to learn both feature extraction and object localization, often achieving superior accuracy compared to classical methods.

Region-based detectors (R-CNN). Region-based Convolutional Neural Networks (R-CNN) [93] and its successors (Fast R-CNN [94], Faster R-CNN [95]) pioneered the use of deep learning for object detection. These models operate in two stages: they first generate region proposals that may contain objects, and then classify and refine bounding boxes for each proposal. While highly accurate, they are often computationally intensive and better suited to natural images with well-separated objects.

One-stage detectors (YOLO, SSD). Single-stage detectors such as YOLO (You Only Look Once) [96] and SSD (Single Shot Multibox Detector) [97] streamline the detection pipeline by directly predicting object classes and bounding boxes. They use a CNN [98] to extract features on a low resolution grid, followed by fully connected layers that predict a fixed number of bounding boxes for each hyper-pixel of the grid. This gives a large (and fixed) number of predicted bounding boxes with several that may fit the same target. Therefore, Non-Maximum Suppression (NMS) is used to remove overlapping bounding boxes with the lowest object probability score. These models are significantly faster and can be deployed in real-time applications. However, because they predict from low resolution grid, they tend to underperform on small or overlapping objects, which are common in microscopy images.

Fully convolutional approaches (U-Net). In bioimaging and dense object detection scenarios, fully convolutional architectures such as U-Net [99] have become the *de facto* standard. U-Net produces pixel-level segmentation maps that can be post-processed into individual de-

tections via connected components or shape fitting. Several adaptations have been proposed to improve instance separation:

- **StarDist** [25] predicts star-convex polygons for each pixel of the image, enabling precise boundary localization even in crowded scenes. It relies on NMS to remove duplicated instances.
- **CellPose** [100] introduces a CNN that learns gradient vector fields pointing toward cell centers, making it robust across a wide range of cell shapes and modalities.
- **OmniPose** [101] extends CellPose by incorporating shape-agnostic flow representations, improving performance on elongated and irregular objects.
- **SpotiFlow** [102] propose an adaption for point-like objects (particles) in large images. It detects particle positions as the local minima in the predicted flow field.

Transformer-based models. Vision Transformers (ViTs) [103, 104] and their detection-specific variants such as DETR (DEtection TRansformer) represent a new direction in object detection. These models replace convolutional backbones with self-attention mechanisms, which can capture long-range dependencies and global context. Although ViTs have demonstrated impressive results on natural images, their adoption in microscopy remains limited due to the amount of data required and training complexity.

Discussion. Despite their effectiveness, deep learning-based detectors present several limitations:

- **Data dependency:** Training high-performing models requires large, accurately annotated datasets, which are often unavailable or expensive to produce in biomedical contexts.
- **Computational cost:** Model training typically demands substantial computational resources (e.g., GPUs), though inference can be fast once the model is trained.
- **Domain generalization:** Models may fail to generalize across domains or imaging conditions not represented in the training data, requiring domain adaptation or fine-tuning.

Nevertheless, with appropriate datasets and careful design, deep learning-based approaches currently deliver the most accurate and robust detection performance across a wide variety of applications in biological imaging.

II.1.3 Linking approaches

II.1.3.1 Graph-based

A first class of linking algorithms solves the tracking problem by formulating it as a graph optimization problem [62, 64, 91, 105–107]. In this formulation, each detection is represented as a node in a graph, and a track $\gamma^i_{1:T}$ is represented as a sequence of edges connecting these nodes over time.

These approaches commonly rely on three main assumptions:

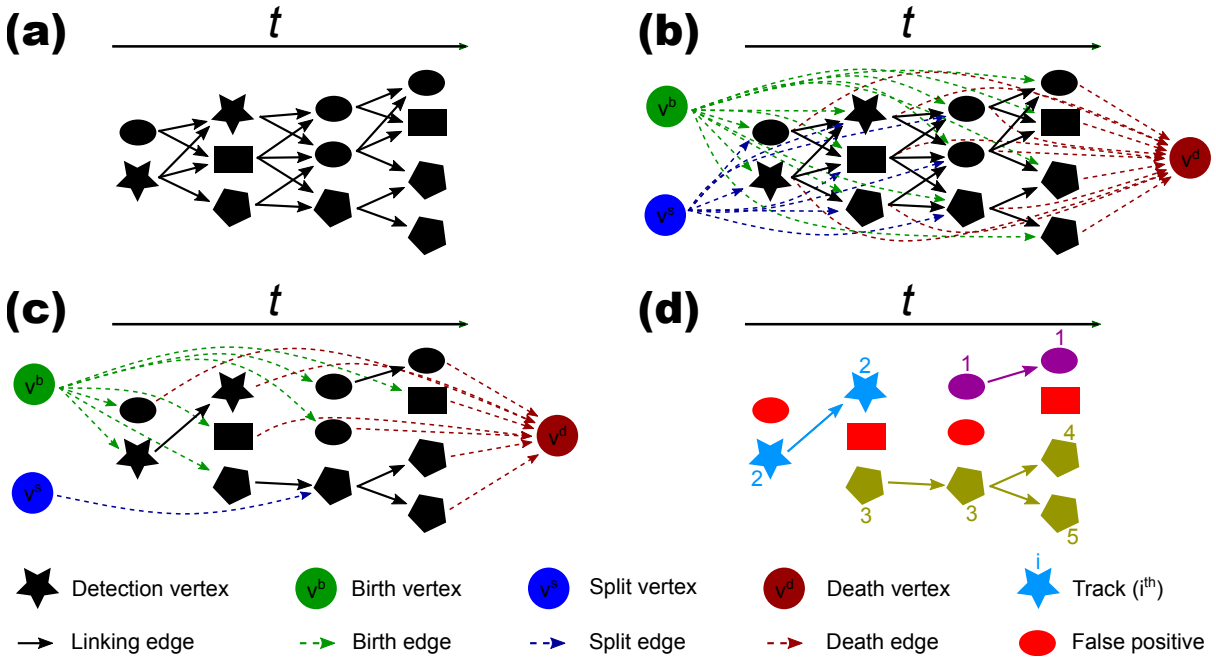


Figure II.3: Graph-based linking approach. Adapted to the tracking example in Figure II.2.a **(a)** A directed acyclic graph G is constructed, where nodes represent detections and edges represent feasible associations. Node shapes are used for visual clarity. **(b)** Special nodes for track birth, death and splitting are added. **(c)** A sub-graph G' is selected to minimize the total cost under structural constraints. A target splitting event occurs at frame 3 for the pentagon-shaped target. **(d)** Reconstructed tracks from the G'. Tracks of length one are considered false positives (in red).

1. **1-to-1 association:** Each target generates at most one detection per frame, and each detection is assigned to at most one track (see Chapter II.1.1).

2. **No-gap:** Tracks are assumed to be continuously detected; i.e., the track $\gamma^i_{1:T}$ is defined on the interval $[\![t^i_{\mathrm{b}}, t^i_{\mathrm{d}}]\!]$ without any missed detections. Therefore in this interval, γ^i_t contains a single (and unique across tracks) detection index denoted j^i_t.

3. **Markovian cost:** The cost of prolonging a track at time t (see Equation II.10) depends only on the immediate previous detection, not the full sequence:

$$c(Z_t^{\gamma^i_t} \mid Z^{\gamma^i_{1:t-1}}) = c(Z_t^{\gamma^i_t} \mid Z^{\gamma^i_{t-1}}) = c(\mathbf{z}_t^{j^i_t} \mid \mathbf{z}_{t-1}^{j^i_{t-1}}). \tag{II.14}$$

This last assumption implies that the prolongation cost is shared among all feasible tracks that include the same pair of consecutive detections, enabling a compact graph representation.

Under these assumptions, all feasible tracks and their associated costs can be represented in a directed acyclic graph (DAG) $G = (V, E)$ (see Figure II.3.a), where each vertex $v^j_t \in V$ corresponds to a detection $\mathbf{z}^j_t \in Z_t$, indexed by frame $t \in [\![1, T]\!]$ and detection index $j \in [\![1, M_t]\!]$. The total number of vertices is equal to the total number of detections:

$$|V| = \sum_{t=1}^{T} M_t. \tag{II.15}$$

Edges are defined between temporally consecutive detections, i.e., for each pair $(\mathbf{z}_t^{j_1}, \mathbf{z}_{t+1}^{j_2})$, an edge $e_t^{j_1,j_2} \in E$ is added with cost $c_t^{j_1,j_2} = c(\mathbf{z}_{t+1}^{j_2} \mid \mathbf{z}_t^{j_1})$. This results in a quadratic number of edges:

$$|E| = \sum_{t=1}^{T-1} M_t M_{t+1}. \tag{II.16}$$

To mitigate computational cost, edges with high cost may be pruned. Nevertheless, the total number of edges scales at best linearly with the number of frames and targets.

To account for track initiation (birth) and termination (death), artificial source and sink nodes $v^{\mathrm{b}}, v^{\mathrm{d}} \in V$ are added (see Figure II.3.b). Each detection is connected to the source via a birth edge with cost c^{b}, and to the sink via a death edge with cost c^{d}. To handle mitosis events (target splitting), a special source node $v^{\mathrm{s}} \in V$ may be introduced, with edges to all other nodes and associated splitting cost c^{s}.

A track $\gamma_{1:T}^i \in \Omega$ is considered feasible if all its detection-to-detection transitions exist in E. A track defines a path in the graph and its total cost is:

$$C(\gamma_{1:T}^i) = c^{\mathrm{b}} + c^{\mathrm{d}} + \sum_{t=t_{\mathrm{b}}^i}^{t_{\mathrm{d}}^i - 1} c_t^{j_t^i, j_{t+1}^i}. \tag{II.17}$$

The linking task (Equation II.8) is reformulated as selecting a subgraph $G' = (V', E')$, with $V' = V$ and $E' \subset E$ (see Figure II.3.c) that minimizes the sum of selected edge costs while respecting structural constraints C. It can be written:

$$\underset{E' \subset E}{\arg\min} \quad \sum_{e_t^{j_1,j_2} \in E'} c_t^{j_1,j_2} \tag{II.18}$$

$$\text{subject to} \quad E' \in \mathcal{C} \tag{II.19}$$

The structural constraints C enforce that each detection node is associated to an unique track (preventing $E = \emptyset$), where a track is a path that starts from the birth node and ends in the death node. The set of tracks $\Gamma_{1:T}^\star$ is recovered as the paths in G' (Figure II.3.d). False positive detections are usually associated to track of length 1, which are dropped.

Graph-constraints and optimization

The structural constraints for G' are typically expressed as either flow conservation [91, 105] or as binary edge classification with linear constraints [62, 64]. We adopt the latter formulation.

Let $\mathbf{y} \in \{0,1\}^{|E|}$ be a binary vector indicating selected edges:

$$e \in E' \iff \mathbf{y}_e = 1 \tag{II.20}$$

where $\mathbf{y}_e$ corresponds to the indicator value associated to the edge $e \in E$.

A valid subgraph G' must satisfy the following constraints [62] defined on the binary vector $\mathbf{y}$:

1. **Unique parent:** Each detection node $v \in V \setminus \{v^{\mathrm{b}}, v^{\mathrm{d}}, v^{\mathrm{s}}\}$ has exactly one incoming edge from another detection node or from v^{b}. By using the convention $\mathbf{y}_{(u,v)} = 0$ when $(u,v) \notin E$, it writes:
$$\mathbf{y}_{(v^{\mathrm{b}},v)} + \sum_{u \in V \setminus \{v^{\mathrm{b}}, v^{\mathrm{d}}, v^{\mathrm{s}}\}} \mathbf{y}_{(u,v)} = 1 \tag{II.21}$$

2. **At most two children:** Each detection node $u \in V \setminus \{v^{\mathrm{b}}, v^{\mathrm{d}}, v^{\mathrm{s}}\}$ has at most two outgoing edges. If none are selected, the death edge (u, v^{d}) must be selected:

$$\frac{1}{2} \sum_{v \in V \setminus \{v^{\mathrm{b}}, v^{\mathrm{d}}, v^{\mathrm{s}}\}} \mathbf{y}_{(u,v)} \leq 1 - \mathbf{y}_{(u,v^{\mathrm{d}})} \leq \sum_{v \in V \setminus \{v^{\mathrm{b}}, v^{\mathrm{d}}, v^{\mathrm{s}}\}} \mathbf{y}_{(u,v)} \tag{II.22}$$

3. **Splitting constraint:** If a node splits into two children, its incoming edge from v^{s} must be selected:

$$\sum_{v \in V \setminus \{v^{\mathrm{b}}, v^{\mathrm{d}}, v^{\mathrm{s}}\}} \mathbf{y}_{(u,v)} - 1 \leq \mathbf{y}_{(v^{\mathrm{s}},u)} \leq \frac{1}{2} \sum_{v \in V \setminus \{v^{\mathrm{b}}, v^{\mathrm{d}}, v^{\mathrm{s}}\}} \mathbf{y}_{(u,v)}. \tag{II.23}$$

Letting $A\mathbf{y} \leq B$ represent the above constraints in matrix form, where A is sparse with $6 \sum_t M_t$ rows (two per constraint per detection) and $|E|$ columns, the linking problem becomes a Mixed Integer Linear Program (MILP):

$$\arg\min_{\mathbf{y}} \mathbf{y}^\top \mathbf{c} \quad \text{subject to } A\mathbf{y} \leq B, \tag{II.24}$$

where $\mathbf{c} \in \mathbb{R}^{|E|}$ is the cost vector of the edges.

This MILP can be (approximately) solved using branch-and-cut algorithms [108], implemented in solvers such as HiGHs [109], SCIP [110], or Gurobi [111]. Notably, this graph formulation drastically reduces the search space compared to general linking optimization: instead of searching in the space of possible track sets $\mathsf{P}(\Omega)$, it searches in the space of edge subsets $\mathsf{P}(E)$, where $|E| \ll |\Omega|$.

Specific graph-based approaches

Tracktour [105] is designed for computational efficiency and relies on a user-in-the-loop paradigm for correcting tracks during post-processing. The edge costs $c_t^{j_1,j_2}$ are defined as the Euclidean distance between the centers of mass of the detections:

$$c_t^{j_1,j_2} = \left\| \mathbf{z}_t^{j_1} - \mathbf{z}_{t+1}^{j_2} \right\|_2 \tag{II.25}$$

The optimization problem is formulated as a Mixed Integer Linear Program (MILP) and solved using Gurobi [111].

Tracktour supports missing detections by allowing merged detections: the structural constraint enforcing a single parent (Equation II.21) is relaxed to allow up to two parents. This enables a track to continue after missing detections by merging temporarily with a neighboring trajectory. However, it relies on the presence of a nearby target and substitutes a missed detection cost with a double linking cost which is inconsistent from a Bayesian perspective (see Chapter II.3.2).

ECLIP [91] models detections as ellipses fitted on their contours and trains a Gradient Boosted Tree (GBT) [112] to predict the cost $c_t^{j_1,j_2}$ of linking two consecutive ellipses. Optimization is again performed using Gurobi.

To address merged detections, ECLIP introduces a hierarchical representation: each detection is decomposed into a set of nested ellipses. Each graph vertex becomes a hyper-vertex composed of one vertex per ellipse. With additional constraints ensuring non-overlapping ellipse selection within a hyper-vertex, Gurobi can jointly solve merged detections and tracking.

Linajea [62] uses a U-Net [99] trained on ground-truth tracks to predict motion vectors $\mathbf{d}_t^j \in \mathbb{R}^d$ for each detection $\mathbf{z}_t^j$. To learn targets dynamics, the motion U-Net incorporates a 7-frame temporal window. The predicted motion vectors estimate the detection's location in the previous frame, enabling motion-corrected edge costs:

$$c_t^{j_1,j_2} = \left\| \mathbf{z}_t^{j_1} - \left(\mathbf{z}_{t+1}^{j_2} - \mathbf{d}_{t+1}^{j_2} \right) \right\|_2 \tag{II.26}$$

In addition, node selection costs are also introduced, proportional to the detection probability estimated by the detector. The total cost of an edge is the sum of its motion-corrected distance and the costs of the associated nodes. Gurobi is used to solve the resulting MILP problem.

Trackastra [64] leverages a Transformer architecture [113] to compute contrastive embeddings $\mathbf{f}_t^j$ for each detection. Initial features $\mathbf{z}_t^j$ encode position, texture, and morphology (e.g., mask area, mean intensity), and are refined through masked attention over a spatial and temporal neighborhood. The final embedding $\mathbf{f}_t^j \in \mathbb{R}^{256}$ encodes contextual information across 6 frames. Association costs are computed using cosine similarity:

$$c_t^{j_1,j_2} = -{\mathbf{f}_t^{j_1}}^\top \mathbf{f}_{t+1}^{j_2} \tag{II.27}$$

Costs are then normalized (see [64]) before being used in a MILP solved by Gurobi (following Linajea [62]).

MAGIK and BGU-IL-5 [106,107] adopt a Graph Neural Network (GNN) architecture [114] trained on ground-truth associations to directly predict edge probabilities $\mathbf{p} \in [0,1]^{|E|}$. A ResNet [115] trained with contrastive learning produces a visual embedding for each detection, which is concatenated with positional features and passed through the GNN with masked attention. Final edge probabilities are obtained through a classification head.

Rather than solving the full MILP–using for instance $c = -\log \mathbf{p}$ as the edge cost–edge selection is performed via thresholding and a greedy heuristic that prioritizes the highest-probability edges. In MAGIK, edges connecting detections up to $\tau_{\max}$ frames apart are added to handle up to $\tau_{\max} - 1$ consecutive missed detections. A greedy algorithm is again used for selecting non-conflicting edges with minimal gap.

Discussion

Graph-based tracking methods formulate the association problem as a global graph optimization task, typically modeled as a MILP. These methods usually rest on three key assumptions:

1. **1-to-1 Association:** Each target generates at most one detection per frame, and each detection is associated with at most one track.
2. **No-Gap:** Targets are continuously detected; missed detections are not explicitly modeled.
3. **Markovian Cost:** Track continuation costs depend only on local information from adjacent frames, not on the track history.

These assumptions significantly reduce the combinatorial complexity of the tracking problem, enabling global optimization. However, they impose important limitations:

1. **Scalability:** With long sequences (>1000 frames) and many targets (>500), the graph size can exceed memory capacity, and MILP solving can become intractable. LINAJEA [62] addresses this by dividing the video into overlapping spatiotemporal blocks with border consistency constraints.

2. **Merged and Split Detections:** The 1-to-1 constraint prevents handling merged or split detections. For instance, in Figure II.3.d, the circle target generated two detections on frame 3, requiring to classify one of them as false positive. However, the assumption can be removed [91, 105, 116, 117] through adequate models: for instance, ECLIP [91] mitigates this by modeling detections as hierarchical ellipses and allowing constrained sub-detection selection.

3. **Missed Detections:** The no-gap assumption causes fragmented tracks when detections are missed. As shown in Figure II.3.d, missing frame 2 causes the track to fragment. MAGIK [107] proposes heuristics to solve the gap-closing directly within the graph (Note that MILP could be used to solve globally the missed detections, but it is rarely done in practice). Gap-closing can also be solved as a post-processing [27, 28, 67].

4. **False Positives:** These are typically handled by discarding short tracks (e.g., of length one), which is suboptimal from a Bayesian perspective (see Chapter II.3.2), where a specific costs should be designed for false positives.

5. **Markovian Limitations:** Ignoring track history leads to suboptimal modeling of target dynamics. This limitation can be mitigated using deep learning, which may implicitly capture temporal patterns.

6. **Supervised Learning Limitations:** Most of these recent methods [62, 64, 91, 106, 107] rely on supervised learning (in particular to reduce the previous Markovian limitations). But this requires annotated data which is often not available in practice. Moreover, linking annotations are designed for ground truth detections. Learning directly on these detections may not generalize well to noisy, real-world detections. TRACKASTRA [64] and MAGIK [107] address this by training on predicted detections or simulating detection noise during training. We also note that some generalization across datasets has been shown in TRACKASTRA [64].

Despite their limitations, graph-based approaches have demonstrated strong performance in recent tracking benchmarks. Notably, methods such as TRACKASTRA (EPFL-CH), TRACKTOUR (MON-AU), and MAGIK (EVIC-ES) achieved competitive results in the Cell Tracking Challenge's Cell Linking Benchmark[1], highlighting the effectiveness of graph-based formulations in complex multi-object tracking scenarios.

II.1.3.2 Viterbi Track Linking

Viterbi Track Linking (VTL) extends graph-based tracking approaches by employing dynamic programming—specifically, the Viterbi algorithm—to iteratively identify optimal paths (tracks) within a tracking graph. The method was initially introduced by Sage et al. [118] for single-target tracking in cluttered environments and was later generalized in BAXTER [119, 120] to support multiple targets, merged detections, and events such as target splitting, birth, and death.

[1] `https://celltrackingchallenge.net/latest-clb-results/`

The underlying graph structure used in VTL is a directed acyclic graph (DAG), similar to the one shown in Figure II.3.b. As in MAGIK [107], additional edges are added to the graph to link detections up to $\tau_{\max}$ frames apart, thereby supporting up to $\tau_{\max} - 1$ consecutive missed detections.

Rather than solving a constrained edge classification problem (discussed in Chapter II.1.3.1), VTL constructs tracks $\Gamma^{\star}_{1:T}$ iteratively. At each iteration, the Viterbi algorithm [121] is used to identify the minimum-cost feasible path which is then added to the set of tracks $\Gamma^{\star}_{1:T}$. Initially, this set is empty. A path (track) is considered feasible if it starts from the birth node, ends in the death node and includes at most one detection per frame, although shared detections between tracks are permitted to model merged detections.

Each added track $\gamma^i_{1:T}$ leads to a change in the global tracking cost $\mathcal{C}$, computed as the difference between the cost of accepting the detections as a track and the cost of treating them as false positives:

$$\Delta\mathcal{C} = C(Z^{\gamma^i_{1:T}}) - \sum_{\mathbf{z}\in Z^{\gamma^i_{1:T}}} c^{\mathrm{f}}(\mathbf{z}) \tag{II.28}$$

The algorithm continues adding tracks as long as $\Delta\mathcal{C} < 0$; that is, as long as the global cost is reduced.

This procedure is a greedy approximation to the global minimum of the linking cost $\mathcal{C}$. Since tracks are added one at a time, globally suboptimal decisions may be made. To mitigate this issue, Baxter introduces swap links: a new track can partially overwrite an existing one if doing so reduces the global cost. In this operation, the former overwritten track is associated to a new minimal ending path.

Track costs

In Baxter, track costs are derived within a Bayesian framework to approximate the Maximum a Posteriori (MAP) probability of the tracks $\Gamma^{\star}_{1:T}$ given the detections $Z_{1:T}$ (see Chapter II.3.2). Let $t^i_1 < \cdots < t^i_{n^i}$ denote the sequence of frames in which track γ^i appears. For notational simplicity, we omit the superscript i and denote the associated detections as $\mathbf{z}^{j_{t_k}}_{t_k}$. The total cost of the track, interpreted as the negative log-posterior, is given by:

$$C(Z^{\gamma_{1:T}}) = -\log p(\gamma_{1:T} \mid Z_{1:T}) = c^{\mathrm{b}}(\mathbf{z}^{j_{t_1}}_{t_1}) + c^{\mathrm{d}}(\mathbf{z}^{j_{t_n}}_{t_n}) + \sum_{k=2}^{n} c(\mathbf{z}^{j_{t_k}}_{t_k} \mid \mathbf{z}^{j_{t_{k-1}}}_{t_{k-1}}) \tag{II.29}$$

Here, c^{b} and c^{d} denote the costs associated with the birth and death events of the track, respectively. The birth cost also integrates the probability of spawning from an existing track (for target splitting). These costs are defined from user-defined priors and may be learned from annotated data. The summation term represents Markovian transition costs that depend only on detections from adjacent frames.

In its original formulation, Baxter assumes a Brownian motion model for target dynamics, resulting in motion costs proportional to the squared Euclidean distance between successive detections. For false positives, a uniform prior over the image space is used, with a cost $c^{\mathrm{f}} = \log|\mathbb{W}|$, where $|\mathbb{W}|$ is the total number of pixels in the image.

Discussion

VTL reformulates the tracking problem as a sequence of path-finding tasks on a directed acyclic graph, solved via dynamic programming. It is based on two key assumptions:

1. **less than 1 detection per target:** Each target generates at most one detection per frame.
2. **Markovian Cost:** Track continuation costs depend only on local information from adjacent frames, not on the track history.

It offers several advantages over traditional graph-based optimization methods by addressing some of their limitations (for instance using a Bayesian framework to support false positives and missed detections). However, VTL remains constrained by a number of assumptions and practical considerations:

1. **Scalability:** Similar to MILP-based graph approaches, VTL requires access to the entire video and graph structure at once, which can be infeasible for long sequences or dense target populations.
2. **Split detections:** While VTL supports merged detections via shared nodes, it does not naturally handle split detections.
3. **Markovian Limitations:** Ignoring track history leads to suboptimal modeling of target dynamics. BAXTER mitigates this limitation in [120], where a Gaussian Mixture Probability Hypothesis Density (GM-PHD) filter is used to first estimate unordered set of target states $X_{1:T}$ from the detections $Z_{1:T}$ (see Chapter II.3.2) before applying VTL to resolve associations.
4. **Suboptimal global optimization:** Because tracks are added greedily and iteratively, global optimality is not guaranteed. Although BAXTER introduces swap links to correct some of the past errors, this mechanism offers only limited corrections.

Despite these limitations, VTL remains a flexible method for object tracking, particularly when integrated with more advanced probabilistic models of target behavior.

II.1.3.3 Online linking

Online linking algorithms address the multi-object tracking problem incrementally, frame by frame, in contrast to the previous offline or batch methods that process the entire video sequence at once. This strategy avoids the high memory and computational complexity associated with full sequence optimization. Instead, associations are made between detections in the current frame and existing track hypotheses, effectively reducing complexity. While this approach sacrifices global optimality, it enables the use of complete track histories to improve local decisions, such as modeling complex target dynamics [27, 53, 57, 63, 122–126].

Let $\Gamma_{1:t}$ denote the set of active tracks up to frame t, with cardinality $n(t) = |\Gamma_{1:t}|$, and let Z_{t+1} represent the set of M_{t+1} detections at time $t+1$. The goal is to update $\Gamma_{1:t}$ to obtain $\Gamma_{1:t+1}$ by associating tracks to detections, initializing new tracks, and pruning unreliable tracks. The online linking process typically involves the following three steps:

1. **Track-to-detection association:** For each track $\gamma^i_{1:t} \in \Gamma_{1:t}$, the algorithm identifies the most likely corresponding detection(s) in Z_{t+1} usually using a motion model and cost minimization. Successfully associated tracks form the subset $\Gamma^{\mathrm{a}}_t \subset \Gamma_{1:t}$, and are extended with the matched detection(s):

$$\forall \gamma^i_{1:t} \in \Gamma^{\mathrm{a}}_t, \quad \gamma^i_{1:t+1} = (\gamma^i_1, \ldots, \gamma^i_t, \gamma^i_{t+1}) \in \Gamma_{1:t+1} \tag{II.30}$$

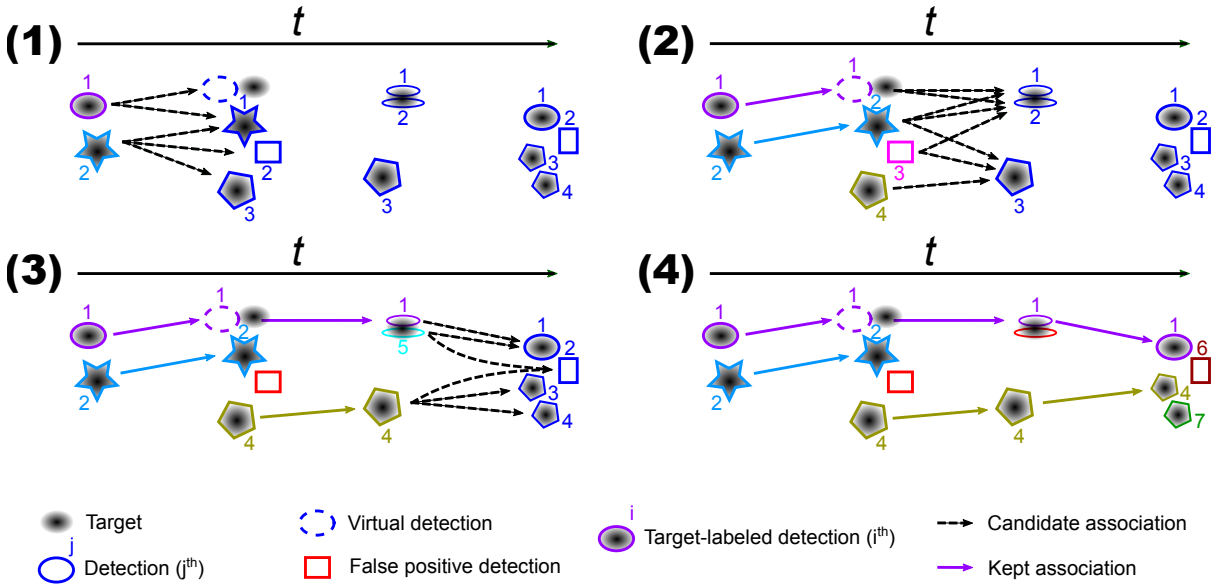


Figure II.4: Illustration of the online linking process, applied to the tracking scenario from Figure II.2.a. The algorithm proceeds iteratively, solving the association frame-by-frame under the 1-to-1 association assumption. Four consecutive frames are shown. **(1)** At initialization, two tracks (1 and 2) are created from the first-frame detections. An association step links these to detections in frame 2. For visualization purposes, a *virtual* (null) detection is shown for track 1, allowing the possibility of no association; in practice, this is considered for all tracks. **(2)** Track 1 is not detected in frame 2 and is propagated with a missing observation ($\gamma_2^1 = \emptyset$), while track 2 is successfully associated. Two new tracks (3 and 4) are initialized from unmatched detections. **(3)** Track 2 is no longer associated and is terminated. Track 3 remains unmatched and is classified as a false positive. Tracks 1 and 4 are correctly propagated. Due to the 1-to-1 constraint, a new track (5) is created to handle the detection split of the circle target. **(4)** Track 5 is unmatched and subsequently removed as a false positive. As the pentagon target undergoes division, track 4 follows one branch, and a new track (7) is created to capture the emerging target. Another track (6) is initialized from the unmatched detection.

2. **Track initialization:** Detections in Z_{t+1} not assigned to any existing track are used to initialize new tracks. For each unmatched detection index $j \in J_{t+1}$, a new track $\gamma_{1:t+1}^{ij}$ is created, with i^j a new track index:

$$\forall j \in J_{t+1} \setminus \bigcup_{\gamma_{1:t}^i \in \Gamma_t^{\mathrm{a}}} \gamma_{t+1}^i, \quad \gamma_{1:t+1}^{ij} = (\emptyset, \ldots, \emptyset, \{j\}) \in \Gamma_{1:t+1} \tag{II.31}$$

3. **Track pruning:** To mitigate the greedy track initialization from the previous frames, non-associated tracks are classified into two disjoint subsets: false tracks Γ_t^{f} and missed tracks Γ_t^{m}, such that:

$$\Gamma_{1:t} = \Gamma_t^{\mathrm{a}} \cup \Gamma_t^{\mathrm{m}} \cup \Gamma_t^{\mathrm{f}}$$

False tracks are removed, while missed tracks are updated with a missing observation:

$$\forall \gamma_{1:t}^i \in \Gamma_t^{\mathrm{m}}, \quad \gamma_{1:t+1}^i = (\gamma_1^i, \ldots, \gamma_t^i, \emptyset) \in \Gamma_{1:t+1} \tag{II.32}$$

The overall algorithm proceeds iteratively for each new frame, starting from an empty set of tracks $\Gamma_{1:0} = \emptyset$. This recursive formulation is summarized in Algorithm 1 and Figure II.4 illustrates how such algorithms work on a small example.

Algorithm 1 Online Linking Algorithm

Require: Sequence of detection sets $\{Z_1, \dots, Z_T\}$
Initialize: $\Gamma_{1:0} \leftarrow \emptyset$
for $t = 0$ to $T - 1$ **do**
  $\Gamma_t^{\mathrm{a}} \leftarrow$ Associate tracks $\Gamma_{1:t}$ to detections Z_{t+1} ▷ Cost modeling & Association
  for all $\gamma_{1:t}^i \in \Gamma_t^{\mathrm{a}}$ **do**
    Extend track: $\gamma_{1:t+1}^i \leftarrow (\gamma_{1:t}^i, \gamma_{t+1}^i)$
  end for

  $\Gamma_{t+1}^{\mathrm{b}} \leftarrow \emptyset$ ▷ Tracks initialization
  for all non-associated detections $\mathbf{z}_{t+1}^j$ **do**
    Create a new track: $\Gamma_{t+1}^{\mathrm{b}} \leftarrow \Gamma_{t+1}^{\mathrm{b}} \cup (\emptyset, \dots \emptyset, \{j\})$
  end for

  Γ_t^{m}, $\Gamma_t^{\mathrm{f}} \leftarrow$ Classify non-associated tracks $\Gamma_{1:t} \setminus \Gamma_t^{\mathrm{a}}$. ▷ Track pruning
  for all $\gamma_{1:t}^i \in \Gamma_t^{\mathrm{m}}$ **do**
    Extend with missing detection: $\gamma_{1:t+1}^i \leftarrow (\gamma_{1:t}^i, \emptyset)$
  end for
  Build the updated tracks set: $\Gamma_{1:t+1} \leftarrow \Gamma_t^{\mathrm{a}} \cup \Gamma_t^{\mathrm{m}} \cup \Gamma_{t+1}^{\mathrm{b}}$
end for
return Final track set $\Gamma_{1:T}$

Track-to-detection association

The global linking objective introduced in Equation II.8 seeks the set of tracks $\Gamma_{1:T}$ that minimizes the global cost $\mathcal{C}(\Gamma_{1:T}, Z_{1:T})$. Online linking methods simplify this optimization by focusing on a frame-by-frame update: at each step, they minimize the additional cost incurred when extending $\Gamma_{1:t}$ to $\Gamma_{1:t+1}$. Under the 1-to-1 association assumption and the cost structure defined in Equation II.10, the cost of associating (prolonging) track $\gamma_{1:t}^i$ with detection $\mathbf{z}_{t+1}^j$ is denoted $\mathbf{C}_{ij} = c(\mathbf{z}_{t+1}^j \mid Z^{\gamma_{1:t}^i})$, which is defined by the linking method.

This association task is commonly formulated as a Linear Assignment Problem (LAP). Formally, we define a bipartite graph $G = (U, V, E)$ where:

- Each track $\gamma_{1:t}^i$ corresponds to a vertex $u_i \in U$.
- Each detection $\mathbf{z}_{t+1}^j \in Z_{t+1}$ corresponds to a vertex $v_j \in V$.
- Edges $E = \{(u_i, v_j)\}_{i,j}$ are weighted by the cost matrix $\mathbf{C}_{ij}$. To ensure computational tractability, edges with high cost may be pruned, reducing to a overall linear number of edges with less than $k \ll M_{t+1}$ most plausible detection candidates per track.

To handle track initiation and termination, virtual nodes and edges $u'_j \in U, (u'_j, v_j) \in E$ and $v'_i \in V, (u_i, v'_i) \in E$ are added, representing track births and deaths (or missed detections), respectively. This leads to an augmented cost matrix $\hat{\mathbf{C}} \in \mathbb{R}^{(n(t)+M_{t+1})\times(n(t)+M_{t+1})}$:

$$\hat{\mathbf{C}} = \begin{pmatrix} \mathbf{C} & \mathbf{C}^{\mathrm{d}} \\ \mathbf{C}^{\mathrm{b}} & \mathbf{0} \end{pmatrix} \tag{II.33}$$

Here, $\mathbf{C}^{\mathrm{b}}$ and $\mathbf{C}^{\mathrm{d}}$ are diagonal matrices encoding birth and death costs, with ∞ used for non-diagonal infeasible assignments.

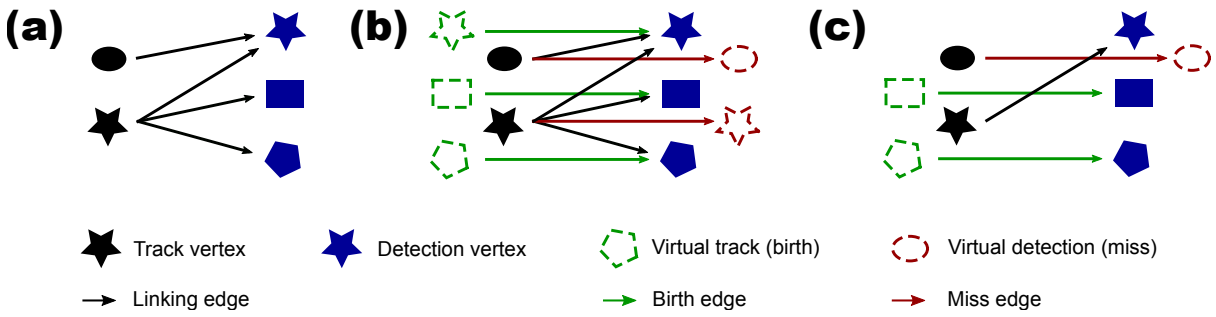


Figure II.5: Linear assignment problem (LAP) applied to the first frame of Figure II.4. **(a)** A bipartite graph is created with the two tracks in $\Gamma_{1:1}$ and the three detections of frame 2. **(b)** Virtual tracks (in green) are added to account for target birth and virtual detections (in red) are added to account for missed detection and target death. **(c)** LAP is solved to find the optimal 1-to-1 association between tracks and detections. In this case, the star target is successfully linked to its detection, the circle target is miss-detected and two new tracks are created (for the rectangle and pentagon detections).

The resulting LAP seeks a binary assignment matrix $\mathbf{Y} \in \{0,1\}^{|U|\times|V|}$ that minimizes total cost while enforcing 1-to-1 matching:

$$\begin{aligned} \text{minimize} \quad & \sum_{i,j} \mathbf{Y}_{ij}\hat{\mathbf{C}}_{ij} \\ \text{subject to} \quad & \sum_{j} \mathbf{Y}_{ij} = 1, \quad \forall i \\ & \sum_{i} \mathbf{Y}_{ij} = 1, \quad \forall j \end{aligned} \tag{II.34}$$

This problem can be solved exactly using the Hungarian algorithm [29] or its faster sparse implementation, the Jonker-Volgenant algorithm [30]. While Mixed Integer Linear Programming (MILP) can also model this, it is generally slower and less efficient for LAP under strict 1-to-1 constraints. To relax these constraints (e.g., to support merging/splitting), heuristic approaches or MILP formulations are used instead [27, 125, 126].

Track management

To maintain a robust track set $\Gamma_{1:t}$ over time, heuristics are commonly employed for handling missed detections and suppressing false positives.

New tracks are initialized from unmatched detections. However, to prevent false positive detections from becoming valid tracks, initiated tracks that are unmatched in their first τ_{min} frames are typically discarded. Values of $\tau_{\text{min}} \in \{2, 3\}$ are commonly used in practice.

To handle missed detections, tracks that are not associated in the current frame are maintained and propagated with empty assignments ($\gamma^i_{t+1} = \emptyset$). These can be re-associated in subsequent frames. However, tracks missing for too long become ambiguous and prone to error. Therefore, a maximum allowed gap τ_{max} is set: if a track is unmatched for more than τ_{max} consecutive frames, it is terminated, and future associations are prohibited by assigning infinite cost: $C_{ij} = \infty$. Common values for τ_{max} range from 1 (no gap) up to 10 depending on the application.

Specific online approaches

FPT [123] is an online linking algorithm based on a custom distance metric that combines spatial and intensity-based features. The assignment problem is addressed using an enhanced greedy optimization strategy. The algorithm begins with a standard greedy solution—selecting at each step the feasible association with the lowest cost—and then iteratively explores transpositions (swaps between assignments) that may further reduce the total cost, until no improvement is possible.

Each segmented detection is represented by a feature vector $\mathbf{z}_t^j = \left(x_t^j, y_t^j, i_t^j, m_t^j\right)^\mathsf{T}$, where (x_t^j, y_t^j) denotes the center of mass, i_t^j is the total intensity (zero-order moment), and m_t^j is the second-order intensity moment. Given a track γ^i associated with detection $\mathbf{z}_t^{j_i}$ on frame t, the cost of linking it to detection $\mathbf{z}_{t+1}^j$ on frame $t+1$ is computed as:

$$\mathbf{C}_{ij} = \|\mathbf{z}_t^{j_i} - \mathbf{z}_{t+1}^j\|_2^2 \tag{II.35}$$

To accommodate missed detections, the algorithm attempt to re-establish links over a temporal window of up to $\tau_{\max}$ frames (typically between 1 and 10, depending on the dataset). However, it does not explicitly handle false positive detections ($\tau_{\min} = 1$).

Huth et al. [53] use a Kalman filter with a constant velocity model to predict the track's position on the next frame. Associations are then computed via a fast approximate nearest-neighbor search using a kd-tree.

Kalman filtering is detailed in section II.3.1.1 and using the same notations, the track's predicted position is given by $\mathbf{HF}\hat{\mathbf{x}}_t^i$, where $\mathbf{F}$ is the transition matrix, $\mathbf{H}$ the measurement matrix, and $\hat{\mathbf{x}}_t^i$ the estimated state on frame t. The cost matrix is computed as:

$$\mathbf{C}_{ij} = \left\|\mathbf{HF}\hat{\mathbf{x}}_t^i - \mathbf{z}_{t+1}^j\right\|_2 \tag{II.36}$$

An additional parameter $\tau_{\max}^{\text{border}} < \tau_{\max}$ is introduced to allow faster track termination near image borders. Mitosis events are handled as target merging events (backward linking), detected by Euclidean distance thresholding.

IML (Instantaneous Maximum Likelihood) [57, 127] employs an Interactive Multiple Model (IMM) framework (detailed in Chapter II.3.1.5), combining three motion models: Brownian (constant position), constant velocity, and constant acceleration. The IMM also models the intensity and size of each target as random variables following Brownian processes.

The IMM allows the estimation of the likelihood $p(\mathbf{z} \mid Z^{\gamma_{1:t}^i})$ of the next detection. The cost matrix is defined to maximize the sum of log-likelihoods as:

$$\mathbf{C}_{ij} = -\log p(\mathbf{z}_{t+1}^j \mid Z^{\gamma_{1:t}^i}) \tag{II.37}$$

Greedy nearest-neighbor matching is used instead of solving the LAP exactly: it selects iteratively the remaining feasible minimal cost.

u-track [27], and its re-implementation **TrackMate** [128] in Fiji [51] propose a two-step LAP framework. Initially, associations are made without support for missed detections ($\tau_{\max} = 1$).

Then, a post-processing step performs tracklet-to-tracklet association, allowing for gap closing, target splitting, and merging. Jonker-Volgenant algorithm is used to solve both LAPs.

The cost matrix can be defined using:

- Euclidean distance for Brownian motion.
- Motion-corrected distance (using a constant velocity Kalman filter) as in Huth et al. [53].
- Intersection-over-union (IoU) for segmented targets that do not move too quickly.

Virtual edge costs ($\mathbf{C}^{\mathrm{b}}$, $\mathbf{C}^{\mathrm{d}}$) are either user-defined or estimated from the past associations. Tracks are only created from at least two consecutive detections ($\tau_{\min} = 2$).

As missed detections are not supported in this linking, it produces *tracklet*, *i.e.* track segments that should be linked globally into full trajectories. A post-processing tracklet-to-tracklet LAP is solved to support the following:

1. **Gap closing:** Tracklets γ^{i_1} and γ^{i_2} can be joined if the temporal gap satisfies $0 < t_{\mathrm{b}}^{i_2} - t_{\mathrm{d}}^{i_1} \leq \tau_{\max}$. The gap closing-cost is the Euclidean distance between the two tracklets.
2. **Target splitting:** A starting tracklet γ^{i_1} may originate from an existing tracklet γ^{i_2}. The splitting cost is the Euclidean distance corrected with an intensity based factor which writes:
$$\rho_{i_1,i_2} = \frac{I_{t-1}(\gamma^{i_2})}{I_t(\gamma^{i_1}) + I_t(\gamma^{i_2})} \tag{II.38}$$
where $I_t(\gamma^i)$ is the sum of the pixel intensities in the image I_t that belongs to track γ^i.
3. **Target merging:** An ending tracklet γ^{i_1} can merge into another existing tracklet γ^{i_2}, with Euclidean distance cost corrected with the factor:
$$\rho'_{i_1,i_2} = \frac{I_{t+1}(\gamma^{i_2})}{I_t(\gamma^{i_1}) + I_t(\gamma^{i_2})} \tag{II.39}$$

Note. Because the second pass operates on the full sequence, the overall pipeline is not truly online.

btrack [124] combines a Bayesian framework with a Kalman filter and a CNN for cell-state classification. Associations are computed using the Jonker-Volgenant algorithm and refined in post-processing optimization, similarly to U-TRACK.

Each detection $\mathbf{z}_t^j$ is classified into cell cycle states s_t^j (e.g., interphase, metaphase) using a CNN. A user-defined state transition matrix encodes biologically plausible state progressions.

The posterior probability for associating detection $\mathbf{z}_{t+1}^j$ to track γ^i can be written:

$$\mathbb{P}(j \in \gamma_{t+1}^i \mid \mathbf{z}_t^j, s_t^j, \gamma_{1:t}^i) \propto p(\mathbf{z}_{t+1}^j \mid Z^{\gamma_{1:t}^i}) \cdot p(s_{t+1}^j \mid s_t^i) \tag{II.40}$$

where $p(\mathbf{z}_{t+1}^j \mid Z^{\gamma_{1:t}^i})$ is given by a constant velocity Kalman filter and $p(s_{t+1}^j \mid s_t^i)$ is the state transition probability.

The Bayesian cost matrix is then defined as:

$$\mathbf{C}_{ij} = -\log \mathbb{P}(j \in \gamma_{t+1}^i \mid \mathbf{z}_t^j, s_t^j, \gamma_{1:t}^i) \tag{II.41}$$

Caliban [126] extends U-TRACK by learning association costs end-to-end from ground-truth tracks.

Each detection $\mathbf{z}_t^j$ is encoded using its 16×16 image patch, centroid position, and handcrafted (texture and morphology) features. A GNN [114] is used to refine the features spatially into $\mathbf{f}_t^j \in \mathbb{R}^d$:

$$(\mathbf{f}_t^j)_j = f^{\text{gnn}}(\mathbf{z}_t^1, \ldots, \mathbf{z}_t^{M_t}) \tag{II.42}$$

Then, for each track $\gamma_{1:t}^i$, an LSTM [49] ingests the past $\tau = 7$ embeddings to encode the temporal evolution and predict:

$$\hat{\mathbf{f}}_{t+1}^i = f^{\text{lstm}}(\mathbf{f}^{\gamma_t^i}, \ldots, \mathbf{f}^{\gamma_{t-\tau+1}^i}) \tag{II.43}$$

Finally, a pairwise head computes the cost through a fully connected layer:

$$\mathbf{C}_{ij} = f^{\text{MLP}}\left(\hat{\mathbf{f}}_{t+1}^i \| \mathbf{f}_{t+1}^j\right). \tag{II.44}$$

Associations are then solved using the same two-step LAP strategy from U-TRACK using the learned cost matrix.

Discussion

The online, frame–by–frame paradigm seeks a *locally* optimal extension of the current track set at every time step. Under the 1-to-1 association assumption this update reduces to a Linear Assignment Problem (LAP), which can be solved exactly in $\mathcal{O}(N^3)$ time with the Hungarian or Jonker–Volgenant algorithms. Because only the present frame and a compact track state must be kept in memory, the approach is computationally light and scales well to long sequences.

A further strength of the online framework is its ability to incorporate explicit motion models: Kalman filters, Interacting Multiple Models, and related Bayesian predictors exploit the full measurement history of each track to generate a prior for the future track position (Chapter II.3.1). In practice this combination of simplicity, efficiency, and model flexibility has proved highly effective. For example, TRACKMATE [128] was used as the baseline for the Cell Tracking Challenge's Cell Linking Benchmark[1] and ranked fourth out of eight competing methods, outperforming several more elaborate, fully supervised pipelines.

However, two main limitations remains:

- **Merged and split detections.** The classical 1-to-1 constraint precludes the direct handling of many-to-one (merged) and one-to-many (split) detection events unless additional heuristics or a second global pass are introduced.
- **Local optimality.** Decisions are taken one frame at a time; consequently, early association errors or false tracks created from spurious detections must be pruned via ad-hoc rules (e.g. thresholds $\tau_{\min}$, $\tau_{\max}$). As a result, the final solution is not guaranteed to minimize the global cost $\mathcal{C}(\Gamma_{1:T}, Z_{1:T})$.

II.1.3.4 Multiple-Hypothesis Tracking (MHT)

Multiple-Hypothesis Tracking (MHT) [26, 129–134] is widely regarded as the gold standard within the tracking-by-detection paradigm. Whereas an online LAP linking commits to a *single* association for every target at each new frame t, MHT *delays* this decision: every active

[1] `https://celltrackingchallenge.net/latest-clb-results/`

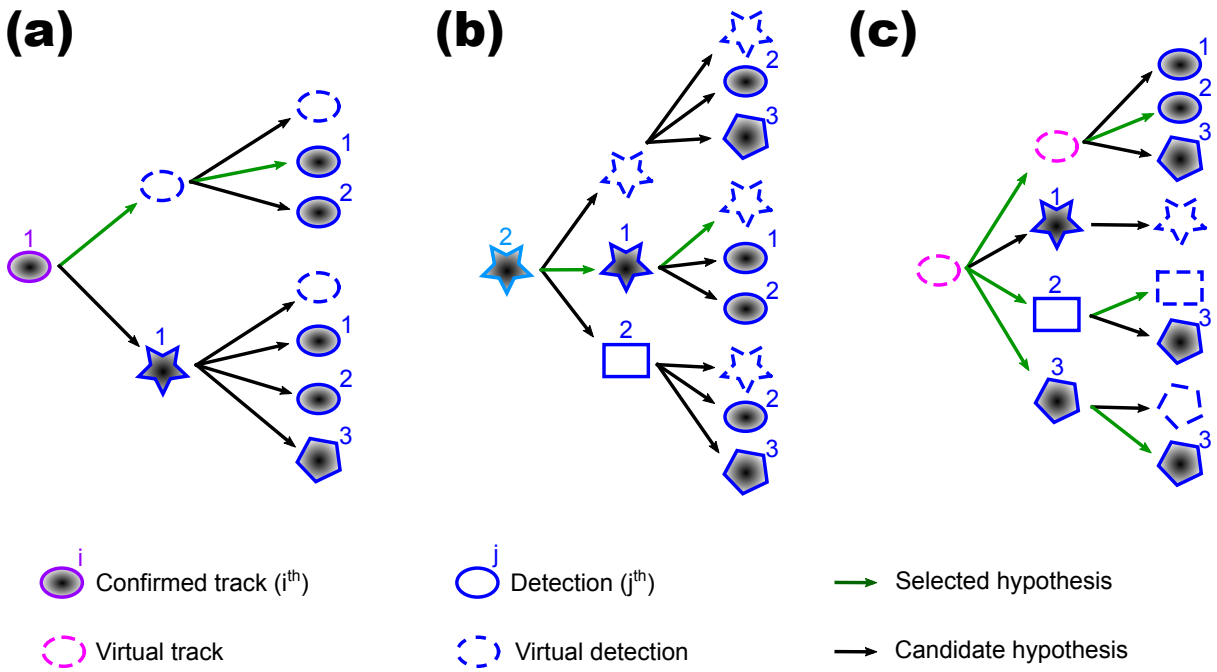


Figure II.6: Illustration of the hypothesis trees in MHT, applied to the tracking scenario from Figure II.2. Hypothesis trees are obtained from frame $t = 1$ with two confirmed tracks (circle and star), and generated with a depth $\tau_{\text{depth}} = 2$. The 1-to-1 optimal association corresponds to the selected hypotheses in green. **(a)** Hypothesis tree for the circle track. With a tree-depth of 2, the association decision is made once the target has re-appeared on frame 3. **(b)** Hypothesis tree for the star track. The track is associated to virtual detection as it terminates on frame 3. **(c)** Track initiation hypotheses, from a virtual track. Two tracks are initiated in frame 2 (rectangle and pentagon). The rectangle track of length one will match with a false positive hypothesis. The 1-to-1 constraint impose to create a track for the split detection of the circle target on frame 3.

track is *branched* into several feasible continuations, referred to as *hypotheses*. Over successive frames each hypothesis is itself propagated, yielding a *hypothesis trees* of depth $1 + \tau_{\text{depth}}$ (see Figure II.6). Because the number of paths grows exponentially, the tree is truncated after at most τ_{depth} frames, and a global optimization—formulated as a Mixed Integer Linear Program (MILP)—is solved to select the subset of paths that minimizes the cost $\mathcal{C}(\Gamma_{1:t+\tau_{\text{depth}}}, Z_{1:t+\tau_{\text{depth}}})$.

The procedure is then advanced by $\tau_{\text{step}} \leq \tau_{\text{depth}}$ frames and repeated. Setting $\tau_{\text{step}} = \tau_{\text{depth}} = T$ yields the globally optimal solution but is computationally infeasible for realistic target counts; typical values are $\tau_{\text{depth}} \in [3, 5]$ and τ_{step}=1. Online LAP tracking can be regarded as the special case $\tau_{\text{depth}} = \tau_{\text{step}} = 1$.

Following the notations developed previously, let $\Gamma_{1:t}$ be the set of confirmed tracks up to frame t. For a look-ahead horizon τ, the set of *feasible tracks* given $\Gamma_{1:t}$ is denoted

$$\Omega(t+\tau \mid \Gamma_{1:t}) = \{\gamma_{1:t+\tau} \in \Omega(t+\tau) \,:\, \gamma_{1:t} \in \Gamma_{1:t} \cup \{(\emptyset, \ldots, \emptyset)\}\},$$

where the all-empty prefix represents a newly-born track between t+1 and t+τ.

At every iteration MHT solves:

$$\Gamma^{\star}_{1:t+\tau_{\text{depth}}} \in \underset{\Gamma_{1:t+\tau_{\text{depth}}} \subset \Omega(t+\tau_{\text{depth}} \mid \Gamma^{\star}_{1:t})}{\arg\min} \mathcal{C}(\Gamma_{1:t+\tau_{\text{depth}}}, Z_{1:t+\tau_{\text{depth}}}) \tag{II.45}$$

To increase the maximal tree depth τ_{depth} feasible with given hardware constraints, it is common to only consider a subset of hypotheses $H_t(\tau) \subset \Omega(t+\tau_{\text{depth}} \mid \Gamma^\star_{1:t})$, typically assuming a 1-to-1 association, and only considering associations with a cost lower than a used-defined threshold [26, 130, 132, 133].

The overall MHT procedure, outlined in Algorithm 2, is divided into two stages. **(1) Hypothesis generation:** a branching stage enumerates candidate prolongations $H_t(\tau)$ and therefore explores the feasible set of tracks $\Omega(t+\tau \mid \Gamma_{1:t})$. **(2) Cost minimization:** a MILP solver chooses, among these candidates, the subset of tracks that minimizes the objective in Equation (II.45). The following sections describe these two components in detail.

Algorithm 2 Multiple-Hypothesis Tracking

Require: Detection sequence $\{Z_1, \dots, Z_T\}$, depth $\tau_{\text{depth}} \leq T$, step $\tau_{\text{step}} \leq \tau_{\text{depth}}$, cost threshold $c_{\max}$.

Initialize: $\Gamma^\star_{1:0} \leftarrow \emptyset, \quad t \leftarrow 0$

while $t < T$ **do** ▷ — Hypothesis generation —
  $H_t(0) \leftarrow \Gamma^\star_{1:t}$
  for $\tau = 0$ **to** $\tau_{\text{depth}} - 1$ **do**
    $H_t(\tau+1) \leftarrow \emptyset$
    for all $\gamma \in H_t(\tau)$ **do**
      for all $\mathbf{z} \in Z_{t+\tau+1}$ **do**
        if $c(\mathbf{z} \mid Z^\gamma) \leq c_{\max}$ **then**
          $H_t(\tau+1) \leftarrow H_t(\tau+1) \cup \big(\gamma, \{\mathbf{z}\}\big)$
        end if
      end for
      $H_t(\tau+1) \leftarrow H_t(\tau+1) \cup \big(\gamma, \emptyset\big)$
    end for
    for all $\mathbf{z} \in Z_{t+\tau+1}$ **do**
      $H_t(\tau+1) \leftarrow H_t(\tau+1) \cup \big(\emptyset, \dots, \emptyset, \{\mathbf{z}\}\big)$
    end for
  end for
  ▷ — Global optimization (MILP) —
  Build cost vector $\mathbf{c}$ and constraint matrix $\mathbf{A}$ from $H_t(\tau_{\text{depth}})$
  $\mathbf{y}^\star \leftarrow \text{MILP\_solve}(\min \mathbf{y}^\top \mathbf{c} \text{ s.t. } \mathbf{A}^\top \mathbf{y} = 1)$
  ▷ — Advance window —
  $\Gamma^\star_{1:t+\tau_{\text{step}}} \leftarrow \{\gamma^k_{1:\tau_{\text{step}}} \mid \gamma^k \in H_t(\tau_{\text{depth}}) \text{ and } \mathbf{y}^\star_k = 1\}$
  $t \leftarrow t + \tau_{\text{step}}$
end while
return $\Gamma^\star_{1:T}$

Hypothesis Generation

Given current tracks $\Gamma_{1:t}$, hypotheses $H_t(\tau) \subset \Omega(t+\tau \mid \Gamma_{1:t})$ are generated recursively to depth τ_{depth}:

$$H_t(0) = \Gamma_{1:t}, \qquad H_t(\tau+1) = \mathcal{G}\big(H_t(\tau)\big), \;\; \tau = 0, \dots, \tau_{\text{depth}} - 1, \tag{II.46}$$

where $\mathcal{G}$ applies three branching rules:

1. **Association:** For every $\gamma_{1:t+\tau} \in H_t(\tau)$ and *feasible* detection $\mathbf{z}^j_{t+\tau+1}$, a prolongation hypothesis is generated:

$$c(\mathbf{z}^j_{t+\tau+1} \mid Z^{\gamma_{1:t+\tau}}) \leq c_{\max} \implies (\gamma_{1:t+\tau}, \{j\}) \in H_t(\tau+1).$$

2. **Miss/death:** For each $\gamma_{1:t+\tau} \in H_t(\tau)$, its empty prolongation is added:

$$(\gamma_{1:t+\tau}, \emptyset) \in H_t(\tau + 1).$$

3. **Birth.** For every detection $\mathbf{z}^j_{t+\tau+1}$ a track birth hypothesis is generated. This is illustrated in Figure II.6.c:

$$(\emptyset, \ldots, \emptyset, \{j\}) \in H_t(\tau + 1).$$

Here, the user-defined threshold $c_{\max}$ prevents the number of hypotheses to grow to quickly. Still, with around p feasible detections per hypothesis, the tree size grows roughly as $N(p+1)^\tau$; hence τ_{depth} is kept small (3 to 5 in practice).

Cost minimization

The optimization aims to find $\Gamma^\star_{1:t+\tau_{\text{depth}}} \subset H_t(\tau_{\text{depth}})$ that minimizes $\mathcal{C}(\Gamma_{1:t+\tau_{\text{depth}}}, Z_{1:t+\tau_{\text{depth}}})$. The cost $\mathcal{C}$ is decoupled into a per track cost C and false positive costs c^{f} following Equation II.10.

Each track hypothesis $\gamma^i_{1:t+\tau_{\text{depth}}} \in H_t(\tau_{\text{depth}})$ comes with a track cost $C(Z^{\gamma^i_{t+\tau_{\text{depth}}}})$. To account for false positive detections, false positive hypotheses are added, such that for each detection $\mathbf{z}^j_t$, a false hypothesis $\gamma^j_{t+\tau_{\text{depth}}} = (\emptyset, \ldots, \emptyset, \{j\}, \emptyset, \ldots, \emptyset)$ is added to $H_t(\tau_{\text{depth}})$, and associated to a cost $c^{\text{f}}(\mathbf{z}^j_t)$.

Formally, let $n = |H_t(\tau_{\text{depth}})|$ be the number of hypotheses considered and $m = |\Gamma^\star_{1:t}| + \sum_{\tau=1}^{\tau_{\text{depth}}} M_{t+\tau}$, the number of confirmed tracks and following detections in the look-ahead horizon τ_{depth}. A binary vector $\mathbf{y} \in \{0,1\}^n$ defines a subset of selected tracks $\Gamma_{1:t+\tau_{\text{depth}}} \subset H_t(\tau_{\text{depth}})$. The selected tracks generate a cost $\mathcal{C}(\Gamma_{1:t+\tau_{\text{depth}}}) = \mathbf{y}^\mathsf{T}\mathbf{c}$, where $\mathbf{c} \in \mathbb{R}^n$ is the cost vector associated with each hypothesis. Finally, a binary matrix $\mathbf{A} \in \{0,1\}^{n\times m}$ is defined such that A_{kl} indicates that the k^{th} hypothesis is using the l^{th} track or detection. The final optimization problem can then be written as:

$$\begin{aligned} &\text{minimize} && \mathbf{y}^\mathsf{T}\mathbf{c} \\ &\text{subject to} && \mathbf{A}^\mathsf{T}\mathbf{y} = \mathbf{1}, \quad \mathbf{y} \in \{0,1\}^n \end{aligned} \tag{II.47}$$

where $\mathbf{A}^\mathsf{T}\mathbf{y} = \mathbf{1}$ is the 1-to-1 association constraint.

Problem (II.47) is a multidimensional LAP and hence a MILP. It is typically solved approximately with branch-and-cut algorithm [108], using solvers such as HiGHs [109], SCIP [110], or Gurobi [111].

Specific MHT approaches

eMHT [26] extends Reid's original Bayesian formulation [130] and is distributed within the Icy platform [50]. Births and false positives are modeled with Poisson priors, while missed detections are handled by a two-state hidden Markov chain that allows every track to enter a "non-perceivable" state. Target motion is predicted via an Interacting Multiple-Model (IMM) filter combining Brownian and constant-velocity dynamics. Track costs are the negative log posterior

$$C(\Gamma_{1:T}, Z_{1:T}) \;=\; -\log\Big[P(\Gamma_{1:T})\, P(Z_{1:T} \mid \Gamma_{1:T})\Big],$$

so minimizing the total cost maximizes the joint posterior. The resulting MILP is solved with a custom branch-and-cut implementation.

MAP [132] replaces the online LAP of IML [57] with an MHT optimization while retaining the same IMM predictor. Probabilistic, *ad-hoc* costs are added for false positives and missed detections. Finally, it proposes to add target splitting and merging hypotheses with specific costs. The method therefore supports cell mitosis. Optimization is carried out with Gurobi [111].

HCRM [133] augments MAP by learning motion likelihoods from annotated data instead of using an IMM. For each hypothetical track $\gamma_{1:t+\tau} \in H_t(\tau)$, a sequence of spatio-temporal features $(\mathbf{f}^i_{t-S+1}, \dots, \mathbf{f}^i_{t+\tau})$ is extracted, where each $\mathbf{f}^i_k$ concatenates the detection coordinates and handcrafted kinematic cues (*e.g.* instantaneous displacement and angle).

A CNN [98] processes this sequence and outputs a data-driven cost (learned on the annotated dataset) for the whole hypothesis, which is then fed to the same MILP solver as in MAP.

BiologicalNeeds [134] builds on EmbedTrack [63]. A CNN trained on pairs of frames jointly predicts detections and per-detection backward motion vectors $\mathbf{d}^j_t \in \mathbb{R}^d$. Positional and motion uncertainties $\mathbf{\Sigma}^{\mathbf{z},j}_t$ and $\mathbf{\Sigma}^{\mathbf{d},j}_t$ are estimated via test-time augmentations.

The likelihood of associating detection $\mathbf{z}^j_t$ to track i is

$$p(\mathbf{z}^j_t \mid Z^{\gamma^i_{1:t-1}}) \;=\; \mathcal{N}(\mathbf{z}^j_t + \mathbf{d}^j_t;\; \mathbf{x}^i_{t-1},\; \mathbf{\Sigma}^i_{t-1} + \mathbf{\Sigma}^{\mathbf{d},j}_t), \tag{II.48}$$

where $\mathbf{x}^i_{t-1}$ and $\mathbf{\Sigma}^i_{t-1}$ corresponds to the position and positional uncertainty of the last associated detections in track i.

This replaces the conventional Bayesian motion model in MHT. The final costs incorporates Bayesian priors on the different events (Poisson for births and false positives, Bernoulli for miss-detected track and death). Additional target splitting hypothesis are generated for mitosis events.

Unlike the Track-Oriented MHT (TO-MHT) framework detailed in this Section, this method adopts the Hypothesis-Oriented MHT (HO-MHT) approach. Instead of maintaining separate hypothesis trees per track, HO-MHT constructs a single global hypothesis tree, where each node represents a complete and feasible set of associations for a given frame. At each time step, new global hypotheses are generated using Murty's algorithm [135,136], which efficiently enumerates the top $A_{\max}$ best-scoring assignment solutions. To manage computational complexity, the hypothesis tree is pruned based on ad-hoc criteria, retaining only the most promising global hypotheses at each iteration.

Discussion

MHT generalizes online (LAP) algorithms by delaying association decisions for τ_{depth} frames, therefore achieving a superior global consistency.

All approaches reviewed here adopt the 1-to-1 association assumption, and therefore do not handle merged or split detections; however, extensions for explicit target splitting and merging have been incorporated into MAP, HCRM and BiologicalNeeds.

A practical limitation of MHT is the exponential growth of the hypothesis tree: depths beyond 3–5 frames quickly become intractable for moderate target counts. For biological applications where targets may vanish for tens of frames (*e.g.* silent Hydra neurons), post-processing can be used to bridge larger gaps (Chapter VI).

II.2 Tracking-By-Regression

Tracking-by-regression refers to a family of approaches that directly estimate the state of targets X_{t+1} in the next frame given their previous states X_t. Unlike the tracking-by-detection paradigm (see Chapter II.1), which decouples detection and linking, these methods tightly couple the prediction of object locations with their spatial and temporal continuity. In this framework, the detection step is implicitly solved through motion estimation.

This paradigm has traditionally been favored in scenarios where the number of objects is known and fixed, and where ground-truth annotations (usually in the first frame) serve as initialization. The task then reduces to propagating object states over time. Recent advances in deep learning have enabled more general usage, where the number of objects is variable and motion models are learned from data.

In the following, we review several representative approaches, from early model-based methods to recent deep-learning solutions. In this thesis, we chose not to use the tracking-by-regression paradigm because of its lack of versatility in many biological applications. Nonetheless, some of its ideas—such as the use of optical flow or deep motion models—can be integrated as priors into tracking-by-detection algorithms.

II.2.1 Active Contours and Deformable Models (Snakes)

Active contour models, or snakes, were among the earliest attempts to perform object tracking by regression. These models evolve an initialized contour around an object to match object boundaries, typically by minimizing an energy functional combining edge and smoothness constraints. Initially developed for refining a coarse segmentation [78] (see Chapter II.1.2.2), they can be extended for tracking, using the contours of the previous frame as initialization.

Zimmer *et al.* [85] extended active contour models based on Mumford–Shah functional [84,87] to the multi-object tracking setting to prevent contours merging in crowded scenarios. Dufour *et al.* [86] further extended this work to 3D volumes by using deformable meshes and integrating a detection module for object initialization. When detections in a new frame do not overlap existing contours, new contours are instantiated.

Discussion. While effective for isolated and slowly moving objects, these methods struggle to track a varying number of targets, especially in the presence of object interactions or topological changes (e.g., splitting or merging) and by construction, struggle to optimize the contour of a target with no overlap between consecutive frames. They also typically require careful initialization and parameter tuning.

II.2.2 Optical Flow-Based Tracking

Optical flow estimates the apparent pixel-wise motion between two consecutive images, allowing any point in the image to be propagated across time. It is widely used in biomechanical displacement analysis [137–139], medical image registration [140], and keypoint tracking [141].

The classical variational formulation seeks a displacement field $\mathbf{\Phi}^\star : \mathbb{R}^d \to \mathbb{R}^d$ such that:

$$I_1(\mathbf{w}) \approx I_2(\mathbf{w} + \mathbf{\Phi}^\star(\mathbf{w})), \tag{II.49}$$

assuming brightness consistency. This leads to a constrained minimization problem:

$$\mathbf{\Phi}^\star = \arg\min_{\mathbf{\Phi}} \|\nabla I_2 \cdot \mathbf{\Phi} + I_2 - I_1\| + R(\mathbf{\Phi}), \tag{II.50}$$

where $R(\mathbf{\Phi})$ is a regularization term encoding smoothness or other priors.

Horn and Schunck [142] proposed the original variational approach using L2 smoothness. Later improvements such as TV-L1 [143, 144] allow for sharper flow discontinuities at object boundaries. To handle larger displacements, pyramidal coarse-to-fine strategies are typically employed [145, 146].

Recent deep learning approaches, such as RAFT [147], formulate flow estimation as a supervised learning problem. Alternatively, unsupervised methods like VoxelMorph [140] learn to register image pairs by minimizing similarity and regularization terms without requiring ground truth flows, typically minimizing the same energy defined by Horn and Schunck.

Discussion. Once computed, optical flow can be used to propagate initialized keypoints across frames. However, this process accumulates errors over time and therefore is less suited for long-term tracking with occlusions or object interactions. Still, they can serve as a valuable motion prior in detection-based linking frameworks [62, 148].

II.2.3 Feature-Based Registration

Feature-based registration approaches aim to track objects by registering an appearance template across frames. These are widely used in single-object tracking (SOT) and visual object tracking (VOT) tasks. A typical example is cross-correlation tracking, as in [149], where a filter is learned to maximize correlation between the template and the target region. Siamese networks such as SiamFC [150] extend this idea to deep features, achieving better invariance to appearance changes.

Multiple improvements have been proposed, including region proposal networks (RPNs) [151] and scale-adaptive architectures [152], allowing tracking of objects undergoing large displacements or size changes.

In the biological domain, ZephIR [153] uses a registration-based approach to track neurons in *Caenorhabditis elegans*. Given annotations in reference frames, target positions are propagated by optimizing a composite loss based on cross-correlation, motion smoothness, and elastic deformation. Although framed as semi-supervised, ZephIR does not involve learning across sequences, but rather performs instance-level optimization.

Discussion. These methods have had limited application in bioimage analysis, largely because biological structures often lack distinctive visual features. Additionally, they typically require a fixed number of targets that are annotated in some reference frames (usually the first frame).

II.2.4 Deep Regression for Multi-Object Tracking

Recent deep learning methods have extended regression-based tracking to multi-object scenarios. Tracktor [154] augments a Faster-RCNN detector by reusing the previous frame's object detections as region proposals for the current frame. The regressor is then applied to refine these proposals, effectively integrating detection and motion modeling.

Transformer-based models such as TrackFormer [60], MOTR [155], and MOTRv3 [156] go further by representing object tracks as query tokens in a transformer architecture. These tokens are updated over time, and the model jointly learns to detect new targets, propagate or terminate existing ones end-to-end. Such models can accommodate varying numbers of targets.

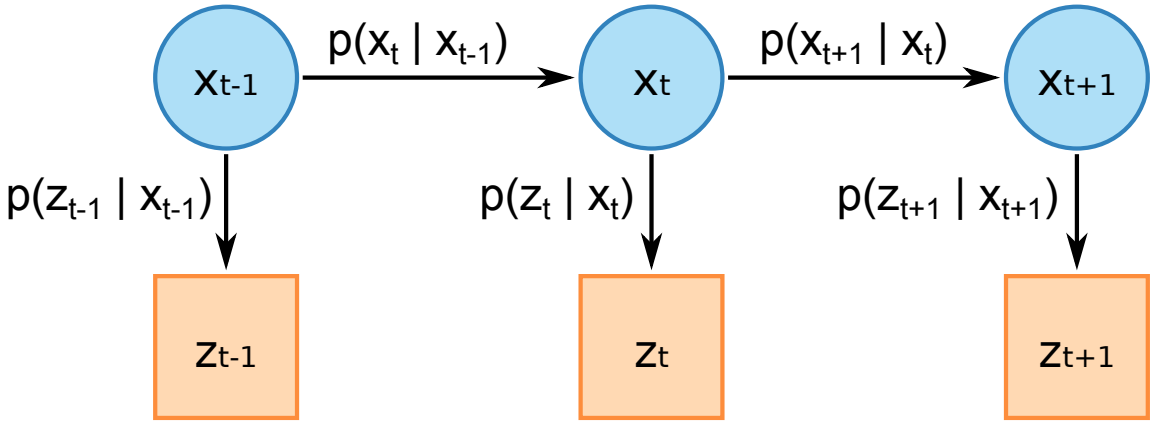


Figure II.7: Hidden Markov Model (HMM) for track states. The state $\mathbf{x}_t \in \mathbb{R}^{d_\mathbf{x}}$ follows a Markov process model that describe the next state given the previous one. A measurement $\mathbf{z}_t \in \mathbb{R}^{d_\mathbf{z}}$ is generated following the measurement model given the current state $\mathbf{x}_t$. Bayesian filtering/smoothing aims at estimating the states while observing only noisy measurements $Z_{1:T} = (\mathbf{z}_1, \ldots, \mathbf{z}_T)$.

Discussion. Despite their potential, these methods have seen limited adoption in bioimage analysis. Biological objects often exhibit subtle visual differences and highly variable dynamics, making it harder to generalize from the limited training data available. Moreover, these models require large annotated datasets and significant computational resources for training.

II.3 Bayesian Framework

This section introduces the Bayesian formulation of multiple-object tracking (MOT). We begin with classical state–space filtering for a *single* target and then extend the formalism to the multi-target case, motivating the tracking-by-detection paradigm and detailing probabilistic cost functions used in in Chapter II.1.

II.3.1 Single-Target State–Space Filtering

When there is just a single target to track, the association is solved: measurements always correspond to this single target at play. Therefore, tracking reduces to a state estimation problem from noisy measurements (with some missing and spurious measurements). In this section, we detail state-space (Bayesian) filtering that addresses the online estimation of an unobserved state vector $\mathbf{x}_t \in \mathbb{R}^{d_\mathbf{x}}$ from a stream of noisy measurements $\mathbf{z}_t \in \mathbb{R}^{d_\mathbf{z}}$. It also predicts the distribution of the future measurement, allowing to build probabilistic association costs, widely used in online linking and MHT linking algorithms. We will also quickly review Bayesian smoothing that solves an offline state estimation of the state $\mathbf{x}_t$ given all the previous and future measurements $Z_{1:T}$. In a tracking-by-detection framework, smoothing is used as a tracking post-processing to improve track estimations once the associations are resolved. The temporal index t is discrete with unit step $(t \mapsto t + 1)$; extension to irregular sampling is straightforward but not required for the present discussion.

Hidden Markov model. The system is modeled as a first-order Hidden Markov Model (HMM) (Figure II.7):

$$\mathbf{x}_t \mid \mathbf{x}_{t-1} \sim p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) \qquad \text{[process model]} \tag{II.51}$$

$$\mathbf{z}_t \mid \mathbf{x}_t \sim p(\mathbf{z}_t \mid \mathbf{x}_t) \qquad \text{[measurement model]} \tag{II.52}$$

where the process and measurement distributions are known (or can be approximated).

Given an initial prior $\mathbf{x}_0 \sim p(\mathbf{x}_0)$, the aim is to compute, for every frame t, the *filtering distribution* $p(\mathbf{x}_t \mid Z_{1:t})$, where $Z_{1:t} = (\mathbf{z}_1, \dots, \mathbf{z}_t)$.

Bayesian recursion. Bayesian filtering proceeds recursively via three steps.

1. **Prediction.** Propagate the previous posterior through the process model to obtain the *prior* at time t:

$$p(\mathbf{x}_t \mid Z_{1:t-1}) = \int p(\mathbf{x}_t \mid \mathbf{x}_{t-1})\, p(\mathbf{x}_{t-1} \mid Z_{1:t-1})\, \mathrm{d}\mathbf{x}_{t-1}. \tag{II.53}$$

2. **Projection.** Project this prior into the measurement space to obtain the future measurement distribution:

$$p(\mathbf{z}_t \mid Z_{1:t-1}) = \int p(\mathbf{z}_t \mid \mathbf{x}_t)\, p(\mathbf{x}_t \mid Z_{1:t-1})\, \mathrm{d}\mathbf{x}_t. \tag{II.54}$$

3. **Update.** Upon observing $\mathbf{z}_t$, the prior is updated to the *posterior* using Bayes' rule:

$$p(\mathbf{x}_t \mid Z_{1:t}) = \frac{p(\mathbf{z}_t \mid \mathbf{x}_t)\, p(\mathbf{x}_t \mid Z_{1:t-1})}{p(\mathbf{z}_t \mid Z_{1:t-1})}. \tag{II.55}$$

   If no measurement is available (*e.g.*, a missed detection) the update step is skipped and the prior becomes the next posterior. When spurious detections are a possibility, a gate on the likelihood $p(\mathbf{z}_t \mid Z_{1:t-1})$ is used to reject an unlikely measurement and treat it as missed.

Equations (II.53)–(II.55) are intractable in the general case. Practical solutions require additional assumptions and approximations: for linear and Gaussian models (II.51)–(II.52), Kalman filtering [157, 158] gives a close forms to the Bayesian recursion. Extended Kalman, Unscented Kalman or particle filters [159–161] approximate the distributions numerically for more general models.

II.3.1.1 Kalman filtering

When both the process and the measurement models are linear and all random disturbances are Gaussian, the Bayesian recursion admits a closed-form analytical solution—the *Kalman filter* [157, 158, 162]. Specifically, let the state–space model be:

$$\mathbf{x}_t = \mathbf{F}\mathbf{x}_{t-1} + \boldsymbol{\epsilon}_t^{\text{mov}} \qquad \boldsymbol{\epsilon}_t^{\text{mov}} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}) \qquad \text{[process]} \tag{II.56}$$

$$\mathbf{z}_t = \mathbf{H}\mathbf{x}_t + \boldsymbol{\epsilon}_t^{\text{det}} \qquad \boldsymbol{\epsilon}_t^{\text{det}} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}) \qquad \text{[measurement]} \tag{II.57}$$

with $\mathbf{F} \in \mathbb{R}^{d_\mathbf{x} \times d_\mathbf{x}}$ the process (or state-transition) matrix and $\mathbf{H} \in \mathbb{R}^{d_\mathbf{z} \times d_\mathbf{x}}$ the measurement matrix. And $\boldsymbol{\epsilon}_t^{\text{mov}}$ (resp. $\boldsymbol{\epsilon}_t^{\text{det}}$) is the process (resp. measurement) random noise vector, which follows independent Gaussian distributions with zero-mean and covariance matrix $\mathbf{Q}$ (resp. $\mathbf{R}$).

Let $\hat{\mathbf{x}}_t$ and $\hat{\mathbf{P}}_t$ denote the mean and the covariance of the filtering distribution. Assuming a Gaussian prior $\mathbf{x}_0 \sim \mathcal{N}(\hat{\mathbf{x}}_0, \hat{\mathbf{P}}_0)$, the filtering distribution remains Gaussian for all t and can be computed recursively:

$$\begin{aligned} \mathbf{x}_t|Z_{1:t-1} &\sim \mathcal{N}(\mathbf{F}\hat{\mathbf{x}}_{t-1}, \mathbf{F}\hat{\mathbf{P}}_{t-1}\mathbf{F}^\mathsf{T} + \mathbf{Q}) \\ &\sim \mathcal{N}(\bar{\mathbf{x}}_t, \bar{\mathbf{P}}_t) \end{aligned} \quad \text{[prediction]} \tag{II.58}$$

$$\begin{aligned} \mathbf{z}_t|Z_{1:t-1} &\sim \mathcal{N}(\mathbf{H}\bar{\mathbf{x}}_t, \mathbf{H}\bar{\mathbf{P}}_t\mathbf{H}^\mathsf{T} + \mathbf{Q}) \\ &\sim \mathcal{N}(\bar{\mathbf{z}}_t, \bar{\mathbf{S}}_t) \end{aligned} \quad \text{[projection]} \tag{II.59}$$

$$\begin{aligned} \mathbf{x}_t|Z_{1:t} &\sim \mathcal{N}(\bar{\mathbf{x}}_t + \mathbf{K}_t(\mathbf{z}_t - \bar{\mathbf{z}}_t), \bar{\mathbf{P}}_t - \mathbf{K}_t\bar{\mathbf{S}}_t\mathbf{K}_t^\mathsf{T}) \\ &\sim \mathcal{N}(\hat{\mathbf{x}}_t, \hat{\mathbf{P}}_t) \end{aligned} \quad \text{[update]} \tag{II.60}$$

where $\mathbf{K}_t = \bar{\mathbf{P}}_t\mathbf{H}^\mathsf{T}\bar{\mathbf{S}}_t^{-1}$ is the *Kalman gain*.

II.3.1.2 Biological Modeling Example

Before exploring more complex filtering approaches, we first illustrate classical biological tracking models employing the Kalman filter (KF). For these examples, we assume a two dimensional tracking problem ($d = 2$), that aims to estimate the position (x_t, y_t) of a target over time. Extensions to three dimensions are straightforward, requiring only the inclusion of an additional spatial dimensions z_t.

In a tracking-by-detection framework, each detection vector $\mathbf{z}_t$ provides a noisy measurement of the true target position at time t. Typically, this is modeled as:

$$\mathbf{z}_t = \begin{pmatrix} x_t & y_t \end{pmatrix}^\mathsf{T} + \boldsymbol{\epsilon}_t^{\text{det}}, \quad \boldsymbol{\epsilon}_t^{\text{det}} \sim \mathcal{N}(\mathbf{0}, \sigma_{\text{det}}^2\mathbf{I}_d), \tag{II.61}$$

where σ_{det} encodes prior knowledge about the localization errors of the detection algorithm. Here, measurement errors are assumed isotropic and uncorrelated, although a general covariance matrix $\mathbf{R}$ may also be employed. Given the linearity and Gaussian assumptions of this measurement model, it naturally fits within the Kalman filtering framework.

We now review several widely used motion models in biological contexts.

Brownian Motion

On subcellular scales, the dynamics of macromolecules, vesicles, viruses, and small organelles are predominantly influenced by thermal fluctuations and viscous drag. These factors make Brownian motion—a Gaussian random walk—an appropriate modeling choice. At larger scales (e.g., cells or small organisms), despite smoother dynamics, low frame-rate imaging (due to microscope constraints, image quality constraints or even data management) often results in incremental target displacements being effectively random. In these scenarios, Brownian motion remains useful, especially when directed forces (chemotaxis, fluid flows) are negligible or absent.

With $\mathbf{x}_t = (x_t, y_t)$ representing the target's position, Brownian motion can be expressed as:

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \boldsymbol{\epsilon}_t^{\text{mov}}, \quad \boldsymbol{\epsilon}_t^{\text{mov}} \sim \mathcal{N}(\mathbf{0}, \sigma_{\text{mov}}^2\mathbf{I}_d), \tag{II.62}$$

where the parameter σ_{mov} controls the diffusivity. This process is linear and Gaussian, thus pairing neatly with the measurement model in Equation (II.61), allowing the use of Kalman filters. Here, the process and measurement matrices simplify to identity matrices, *i.e.*, $\mathbf{F} = \mathbf{H} = \mathbf{I}_d$.

The Kalman filter, in this case, effectively provides a denoised position estimate and prediction of future measurements. Specifically, the predicted measurement at the next time point is a weighted moving average of previous measurements, where the weights depends on the Kalman gain K_t:

$$\bar{\mathbf{z}}_{t+1} = \mathbf{HF}\hat{\mathbf{x}}_t = \hat{\mathbf{x}}_t \tag{II.63}$$
$$= \mathbf{K}_t\mathbf{z}_t + (1 - \mathbf{K}_t)\bar{\mathbf{z}}_t \tag{II.64}$$

Higher-Order Motion Models

For larger biological objects (cells, small organisms) or smaller targets driven by external forces (fluid flow, tissue deformation), motion can exhibit more structure, allowing the use of higher-order motion models that incorporate velocity and acceleration.

For instance, in a *constant velocity model* [27, 53, 162], the state is extended to $\mathbf{x}_t = (x_t, \dot{x}_t, y_t, \dot{y}_t)^\mathsf{T}$, assuming constant velocity between frames, perturbed by Gaussian noise. The corresponding linear Gaussian process model is given by the process matrices [162]:

$$\mathbf{F} = \begin{pmatrix} 1 & dt & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & dt \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad \mathbf{Q} = \sigma^2_{\text{mov}} \begin{pmatrix} \frac{dt^4}{4} & \frac{dt^3}{2} & 0 & 0 \\ \frac{dt^3}{2} & dt^2 & 0 & 0 \\ 0 & 0 & \frac{dt^4}{4} & \frac{dt^3}{2} \\ 0 & 0 & \frac{dt^3}{2} & dt^2 \end{pmatrix} \tag{II.65}$$

Similarly, for a *constant acceleration model* [122,162], acceleration terms $(\ddot{x}_t, \ddot{y}_t)$ are included, resulting in a six-dimensional state with a corresponding process matrix $\mathbf{F}$ and covariance matrix $\mathbf{Q}$ provided by:

$$\mathbf{F} = \begin{pmatrix} 1 & dt & \frac{dt^2}{2} & 0 & 0 & 0 \\ 0 & 1 & dt & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & dt & \frac{dt^2}{2} \\ 0 & 0 & 0 & 0 & 1 & dt \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad \mathbf{Q} = \sigma^2_{\text{mov}} \begin{pmatrix} \frac{dt^6}{36} & \frac{dt^5}{12} & \frac{dt^4}{6} & 0 & 0 & 0 \\ \frac{dt^5}{12} & \frac{dt^4}{4} & \frac{dt^3}{2} & 0 & 0 & 0 \\ \frac{dt^4}{6} & \frac{dt^3}{2} & dt^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{dt^6}{36} & \frac{dt^5}{12} & \frac{dt^4}{6} \\ 0 & 0 & 0 & \frac{dt^5}{12} & \frac{dt^4}{4} & \frac{dt^3}{2} \\ 0 & 0 & 0 & \frac{dt^4}{6} & \frac{dt^3}{2} & dt^2 \end{pmatrix} \tag{II.66}$$

In both these cases, past measurements $\mathbf{z}_{1:t}$ are used to estimate a state $\mathbf{x}_t$ (position and higher order dynamics), which is used to predict the future position of the target (x_{t+1}, y_{t+1}). When the near constant velocity (or acceleration) is verified, the predicted position is much more precise that the one from a Brownian Kalman filter.

Modeling Additional Features

Bayesian filtering can also incorporate the evolution of target attributes beyond position, such as shape or average intensity [57, 123]. Typically, these attributes evolve independently from position dynamics and can be modeled using Brownian or constant-velocity assumptions.

II.3.1.3 Non-linear filtering

When either the process model or the measurement model is non-linear, the standard Kalman filter is no longer applicable. In such cases, non-linear extensions like the Extended Kalman

Filter (EKF) [159] and the Unscented Kalman Filter (UKF) [160] are commonly used. These methods aim to approximate the Bayesian filtering equations under non-linear dynamics. The EKF linearizes the models locally via a first-order Taylor expansion, while the UKF relies on deterministic sampling (unscented transform) to better capture non-linearities in the mean and covariance.

Despite their theoretical relevance, non-linear Kalman filters are rarely used in biological tracking because both the motion (e.g., Brownian or directed motion) and the measurement processes (e.g., center of mass localization) are linear models.

A typical example of a non-linear measurement model would be the measurement of a bearing and range in polar coordinates: $\mathbf{z}_t = (\theta_t, r_t)$, where $\theta_t = \arctan(y_t/x_t)$ and $r_t = \sqrt{x_t^2 + y_t^2}$. Such models appear in radar tracking or robotics but are uncommon in microscopy, where measurements usually consist of pixel-localized intensities in Cartesian coordinates.

II.3.1.4 Particle filtering

Particle filtering [161,163,164], also known as Sequential Monte Carlo (SMC) methods, is a powerful family of algorithms for approximating the filtering distribution $p(\mathbf{x}_t \mid Z_{1:t})$ in state-space models, especially when neither the process nor the measurement models are linear or Gaussian. Unlike Kalman filters, which represent the state distribution by its mean and covariance, particle filters use a set of weighted samples (particles) to approximate arbitrary probability distributions.

Each particle represents a hypothetical state, propagated through a stochastic process model. Upon receiving a new measurement, the particles are re-weighted according to their likelihood under the measurement model. A resampling step is typically performed to prevent degeneracy, where most particles have negligible weight.

Particle filters are particularly advantageous in scenarios with:

- Non-linear and/or non-Gaussian dynamics.
- Non-symmetric or multi-modal state distribution.

Despite their flexibility, particle filters are rarely used in biological tracking applications. This is primarily because in biology both motion and measurement models are well-approximated by linear-Gaussian models, for which Kalman filters provide a faster and analytically tractable solution. Additionally, it has a much higher computational cost that scales poorly with the state dimension and the number of tracked targets.

Nonetheless, several studies have demonstrated that particle filters could model the image generation process, where the measurement is directly the image I_t itself. It therefore enables tracking without the need for a prior and decoupled detection step [58, 165, 166]. We also note that with linear and Gaussian models, particle filters are useful to track at once multiple targets with a single multi-modal state, where each mode corresponds to a target, this will be developed in Chapter II.3.2.

II.3.1.5 Interactive Multiple Models (IMM)

As described in Chapter II.3.1.2, biological targets may exhibit different motion patterns depending on their type, location, or time-dependent behavior. While a single motion model (e.g.,

Brownian or constant velocity) may suffice for certain conditions, several limitations arise in practical applications:

1. Different targets in the same video may experience distinct motion regimes, due to spatial heterogeneity in the force field (e.g., tissue deformation, local fluid flow).
2. A single target may switch between different motion regimes during its lifespan, such as transitioning from random diffusion to directed motion under active transport.
3. For non-expert users, selecting a suitable motion model a priori can be difficult or even impossible without domain knowledge.

The *Interactive Multiple Model* (IMM) [167] filter addresses these challenges by combining multiple motion models in a probabilistic framework. Instead of committing to a single model, IMM maintains a set of K candidate motion models $\mathcal{M}^{(k)}$ (e.g., Brownian, constant velocity, accelerated), each associated with its own Kalman filter. The IMM recursively estimates the state $\mathbf{x}_t$ and its model probabilities $\mu_t^{(k)}$ using a mixture of filters.

IMM formulation. Let each model $\mathcal{M}_k$ define:

- A process model: $\mathbf{x}_t = \mathbf{F}^{(k)}\mathbf{x}_{t-1} + \boldsymbol{\epsilon}_t^{\text{mov},(k)}$, with $\boldsymbol{\epsilon}_t^{\text{mov},(k)} \sim \mathcal{N}(0, \mathbf{Q}^{(k)})$.
- A measurement model: $\mathbf{z}_t = \mathbf{H}^{(k)}\mathbf{x}_t + \boldsymbol{\epsilon}_t^{\text{det},(k)}$, with $\boldsymbol{\epsilon}_t^{\text{det},(k)} \sim \mathcal{N}(0, \mathbf{R}^{(k)})$.

Let $\hat{\mu}_t^{(k)} = \mathbb{P}(\mathcal{M}_t^{(k)}|Z_{1:t})$ denote the posterior filtering probability of model k being active at time t, the IMM recursively estimates the state mean $\hat{\mathbf{x}}_t$ and its covariance $\hat{\mathbf{P}}_t$ as follows:

1. Mixing (Interaction): Given model probabilities $\hat{\mu}_{t-1}^{(j)}$ and a Markovian transition matrix $\pi \in \mathbb{R}^{K\times K}$, where $\pi_{kj} = \mathbb{P}(\mathcal{M}_t^{(k)} \mid \mathcal{M}_{t-1}^{(j)})$, compute the mixed initial condition for each model:

$$\tilde{\mu}_{t-1}^{(k|j)} = \frac{\pi_{kj}\hat{\mu}_{t-1}^{(j)}}{\sum_{i=1}^{K}\pi_{ki}\hat{\mu}_{t-1}^{(i)}} \tag{II.67}$$

$$\tilde{\mathbf{x}}_{t-1}^{(k)} = \sum_{j=1}^{K}\tilde{\mu}_{t-1}^{(k|j)}\hat{\mathbf{x}}_{t-1}^{(j)} \tag{II.68}$$

$$\tilde{\mathbf{P}}_{t-1}^{(k)} = \sum_{j=1}^{K}\tilde{\mu}_{t-1}^{(k|j)}\left[\hat{\mathbf{P}}_{t-1}^{(j)} + (\hat{\mathbf{x}}_{t-1}^{(j)} - \tilde{\mathbf{x}}_{t-1}^{(k)})(\hat{\mathbf{x}}_{t-1}^{(j)} - \tilde{\mathbf{x}}_{t-1}^{(k)})^{\mathsf{T}}\right] \tag{II.69}$$

2. Model-specific filtering: Run a standard Kalman filter predict, project and update for each model $\mathcal{M}^{(k)}$ using the mixed state $(\tilde{\mathbf{x}}_{t-1}^{(k)}, \tilde{\mathbf{P}}_{t-1}^{(k)})$ as the previous posterior:

$$\bar{\mathbf{x}}_t^{(k)} = \mathbf{F}^{(k)}\tilde{\mathbf{x}}_{t-1}^{(k)} \qquad \bar{\mathbf{P}}_t^{(k)} = \mathbf{F}^{(k)}\tilde{\mathbf{P}}_{t-1}^{(k)}{\mathbf{F}^{(k)}}^{\mathsf{T}} + \mathbf{Q}^{(k)} \quad \text{[prediction]} \tag{II.70}$$

$$\bar{\mathbf{z}}_t^{(k)} = \mathbf{H}^{(k)}\bar{\mathbf{x}}_t^{(k)} \qquad \bar{\mathbf{S}}_t^{(k)} = \mathbf{H}^{(k)}\bar{\mathbf{P}}_t^{(k)}{\mathbf{H}^{(k)}}^{\mathsf{T}} + \mathbf{R}^{(k)} \quad \text{[projection]} \tag{II.71}$$

$$\hat{\mathbf{x}}_t^{(k)} = \bar{\mathbf{x}}_t^{(k)} + \mathbf{K}_t^{(k)}(\mathbf{z}_t - \bar{\mathbf{z}}_t^{(k)}) \qquad \hat{\mathbf{P}}_t^{(k)} = \bar{\mathbf{P}}_t^{(k)} - \mathbf{K}_t^{(k)}\bar{\mathbf{S}}_t^{(k)}{\mathbf{K}_t^{(k)}}^{\mathsf{T}} \quad \text{[update]} \tag{II.72}$$

3. Model likelihood and update: Compute the likelihood of the measurement $\mathbf{z}_t$ under each model:

$$\Lambda_t^{(k)} = p(\mathbf{z}_t \mid \mathcal{M}_t^{(k)}, Z_{1:t-1}) = \mathcal{N}(\mathbf{z}_t; \bar{\mathbf{z}}_t^{(k)}, \bar{\mathbf{S}}_t^{(k)}) \tag{II.73}$$

Update the model probabilities via Bayes' rule:

$$\hat{\mu}_t^{(k)} = \frac{\Lambda_t^{(k)} \sum_{j=1}^{K} \pi_{kj} \hat{\mu}_{t-1}^{(j)}}{\sum_{i=1}^{K} \Lambda_t^{(i)} \sum_{j=1}^{K} \pi_{ij} \hat{\mu}_{t-1}^{(j)}} \tag{II.74}$$

4. State fusion: Compute the final state estimate as the weighted mixture of the model-specific estimates:

$$\hat{\mathbf{x}}_t = \sum_{k=1}^{K} \mu_t^{(k)} \hat{\mathbf{x}}_t^{(k)} \tag{II.75}$$

$$\hat{\mathbf{P}}_t = \sum_{k=1}^{K} \mu_t^{(k)} \left[\hat{\mathbf{P}}_t^{(k)} + (\hat{\mathbf{x}}_t^{(k)} - \hat{\mathbf{x}}_t)(\hat{\mathbf{x}}_t^{(k)} - \hat{\mathbf{x}}_t)^{\mathsf{T}} \right] \tag{II.76}$$

Discussion. IMM enables flexible, adaptive tracking without requiring manual model selection. It is particularly useful when different targets (or a single target across time) exhibit distinct types of motion. For instance, in microscopy data, vesicles may switch between diffusive wandering and active transport. By combining a Brownian and constant-velocity model, IMM automatically adjusts its estimates based on observed dynamics.

IMM has been successfully applied in biological tracking applications such as IML [57, 127], EMHT [26] and MAP [132], where multiple models are used to capture the diversity of intracellular motion patterns. Despite slightly increased computational complexity (running K filters instead of one), IMM remains tractable and interpretable, making it a practical extension of Kalman filtering in real-world scenarios.

II.3.1.6 State-Space Smoothing

Bayesian filtering, as discussed in Chapter II.3.1, provides an *online* estimate of the hidden state $\mathbf{x}_t$ based only on the sequence of measurements up to time t, yielding the filtering distribution:

$$p(\mathbf{x}_t \mid Z_{1:t}). \tag{II.77}$$

This is well suited for online linking algorithms, where decisions are made in real time. However, in many tracking applications—particularly offline settings—the full sequence of measurements $Z_{1:T}$ is available, and the goal is to estimate the state $\mathbf{x}_t$ using *all* observations across time. This leads to the *smoothing* distribution:

$$p(\mathbf{x}_t \mid Z_{1:T}), \tag{II.78}$$

which is expected to be more accurate than the filtering estimate, especially in the presence of ambiguous, noisy, or missing observations.

Bayesian smoothing. The general Bayesian smoothing recursion can be derived from the filtering distributions $p(\mathbf{x}_t \mid Z_{1:t})$ using a backward recursion. First, one can observe that the

filtering and smoothing distribution are equals for $t = T$. Then, for $t = T-1, \ldots, 1$, the smoothing distribution is obtained via the integral:

$$p(\mathbf{x}_t \mid Z_{1:T}) = \int p(\mathbf{x}_t \mid \mathbf{x}_{t+1}, Z_{1:t})\, p(\mathbf{x}_{t+1} \mid Z_{1:T})\, d\mathbf{x}_{t+1}. \tag{II.79}$$

where the conditional term $p(\mathbf{x}_t \mid \mathbf{x}_{t+1}, Z_{1:T})$ can be expressed using Bayes' rule and the Markov property:

$$p(\mathbf{x}_t \mid \mathbf{x}_{t+1}, Z_{1:T}) = p(\mathbf{x}_t \mid \mathbf{x}_{t+1}, Z_{1:t}) = \frac{p(\mathbf{x}_{t+1} \mid \mathbf{x}_t)\, p(\mathbf{x}_t \mid Z_{1:t})}{p(\mathbf{x}_{t+1} \mid Z_{1:t})}, \tag{II.80}$$

allowing Equation (II.79) to be written entirely in terms of the process model and the filtering estimates.

Intractability. In general, the integral in Eq. (II.79) is not analytically tractable, especially in nonlinear or non-Gaussian systems. In such cases, numerical approximations (e.g., particle smoothing, extended Kalman smoothing) are needed.

However, in the case of linear and Gaussian state-space models (as assumed in the Kalman filter), the smoothing recursion admits a closed-form solution: the Rauch–Tung–Striebel (RTS) smoother [168].

Rauch–Tung–Striebel (RTS) Smoother

Let $\hat{\mathbf{x}}_t$ and $\hat{\mathbf{P}}_t$ denote the filtering mean and covariance obtained from the Kalman filter at time t, and let $\bar{\mathbf{x}}_{t+1}$ and $\bar{\mathbf{P}}_{t+1}$ denote the predicted mean and covariance at time $t+1$ (see Eq. (II.70)). The RTS smoother recursively computes the smoothed estimates $\hat{\mathbf{x}}_t^s$ and $\hat{\mathbf{P}}_t^s$ as follows:

Initialization (at $t = T$):

$$\hat{\mathbf{x}}_T^s = \hat{\mathbf{x}}_T, \qquad \hat{\mathbf{P}}_T^s = \hat{\mathbf{P}}_T. \tag{II.81}$$

Backward recursion (for $t = T-1, \ldots, 1$):

$$\mathbf{G}_t = \hat{\mathbf{P}}_t \mathbf{F}^\mathsf{T} \bar{\mathbf{P}}_{t+1}^{-1} \tag{II.82}$$

$$\hat{\mathbf{x}}_t^s = \hat{\mathbf{x}}_t + \mathbf{G}_t \left(\hat{\mathbf{x}}_{t+1}^s - \bar{\mathbf{x}}_{t+1}\right) \tag{II.83}$$

$$\hat{\mathbf{P}}_t^s = \hat{\mathbf{P}}_t + \mathbf{G}_t \left(\hat{\mathbf{P}}_{t+1}^s - \bar{\mathbf{P}}_{t+1}\right) \mathbf{G}_t^\mathsf{T} \tag{II.84}$$

where $\mathbf{G}_t$ is the *smoothing gain*, analogous to the Kalman gain used in filtering. This backward pass adjusts the filtered estimate at time t using the difference between the smoothed and predicted values at time $t+1$.

Interpretation. Smoothing retroactively improves past estimates based on future observations. It is especially useful when:

- Observations are noisy or missing intermittently.
- The dynamics of the system are weakly constrained (e.g., Brownian motion).
- One desires a refined, smooth trajectory reconstruction for downstream tasks (e.g., kinematic analysis, behavioral modeling).

Although smoothing cannot be used in real-time systems, it is commonly employed as a post-processing step in multiple-object tracking pipelines.

Extensions. For nonlinear systems, the RTS smoother can be extended to the Extended Kalman Smoother (EKS) or Unscented Kalman Smoother (UKS), or approximated with particle-based methods. However, these are less common in biological applications, where the assumption of linear-Gaussian dynamics is often sufficient.

II.3.2 Bayesian Multiple Object Tracking

In Bayesian Multiple Object Tracking, each target i is modeled by a hidden state vector $\mathbf{x}_t^i \in \mathbb{R}^{d_\mathbf{x}}$ that evolves over time. The total number of targets N is unknown and must be inferred jointly with their states. The goal of a Bayesian tracker is to estimate the most probable joint sequence of states $X_{1:T} = (\mathbf{x}_t^i)_{1\leq i\leq N,\, 1\leq t\leq T}$ given the observed data. This leads to the Maximum A Posteriori (MAP) estimation:

$$X_{1:T}^\star \in \arg\max_{X_{1:T}} \mathbb{P}(X_{1:T} \mid I_{1:T}) \tag{II.85}$$

Using Bayes' rule, this posterior can be decomposed as:

$$X_{1:T}^\star \in \arg\max_{X_{1:T}} \mathbb{P}(X_{1:T})\mathbb{P}(I_{1:T} \mid X_{1:T}) \tag{II.86}$$

Here, $\mathbb{P}(X_{1:T})$ encodes prior knowledge about the number of targets N, target birth and death processes, splitting or merging events, and the expected spatial and temporal behavior (e.g., targets may preferentially be localized in the image center). The likelihood $\mathbb{P}(I_{1:T} \mid X_{1:T})$ models the probability of observing the sequence of images $I_{1:T}$ given the states.

In practice, modeling the full image generation process ($\mathbb{P}(I_{1:T} \mid X_{1:T})$) is extremely challenging due to background clutter, imaging noise, occlusions, texture variations, and depth information, which may not be modeled in the state vectors. Some works such as [58, 165, 166] attempt to address this by modeling targets as Gaussian spots with additive background, but such methods remain rare and are usually limited to Gaussian-shape targets without structured background.

Instead, most practical Bayesian tracking algorithms adopt a *tracking-by-detection* paradigm, wherein a detector is first applied to the images to extract a set of measurements:

$$Z_{1:T} = \text{Detector}(I_{1:T}) \tag{II.87}$$

Each Z_t is an arbitrarily ordered set of detections $\mathbf{z}_t^j \in \mathbb{R}^{d_\mathbf{z}}$, as described in Chapter II.1. The detections then serve as proxies for the observations, and the Bayesian tracking problem becomes:

$$\begin{aligned} X_{1:T}^\star &\in \arg\max_{X_{1:T}} \mathbb{P}(X_{1:T} \mid Z_{1:T}) \\ &\in \arg\max_{X_{1:T}} \mathbb{P}(X_{1:T})\mathbb{P}(Z_{1:T} \mid X_{1:T}) \end{aligned} \tag{II.88}$$

Even with this simplification, evaluating the likelihood $\mathbb{P}(Z_{1:T} \mid X_{1:T})$ remains difficult, as it requires to model the measurement process (the detector) and resolving ambiguous data associations — i.e., determining which detection corresponds to which target.

To mitigate this, most approaches introduce a latent association variable $\Gamma_{1:T}$, as defined in Chapter II.1.1, which encodes the assignment of detections to tracks. The MAP estimation is then decoupled into two subproblems:

1. **Association estimation:**

$$\begin{aligned}\Gamma^{\star}_{1:T} &\in \arg\max_{\Gamma_{1:T}} \mathbb{P}(\Gamma_{1:T} \mid Z_{1:T}) \\ &\in \arg\max_{\Gamma_{1:T}} \mathbb{P}(\Gamma_{1:T})\mathbb{P}(Z_{1:T} \mid \Gamma_{1:T})\end{aligned} \tag{II.89}$$

2. **State estimation:**

$$X^{\star}_{1:T} \in \arg\max_{X_{1:T}} \mathbb{P}(X_{1:T} \mid \Gamma^{\star}_{1:T}, Z_{1:T}) \tag{II.90}$$

Although this decoupling is suboptimal in theory (the joint MAP estimate is not recovered), it is computationally tractable and performs well in practice. Recent Random Finite Sets (RFS) [169, 170] models can be used to directly tackle the joint MAP estimate (see Chapter II.3.3).

Association Estimation. Solving Equation (II.89) requires modeling the prior $\mathbb{P}(\Gamma_{1:T})$, which encodes our assumptions about the detection process (*e.g.*, false positive and missed detections) and target events (e.g., birth, splitting, merging and death processes). In practice, many approaches simplify the objective to a likelihood maximization one, ignoring the prior. However, this amounts to assume a uniform prior $\mathbb{P}(\Gamma_{1:T}) = \text{cte}$ which is an unrealistic assumption: in most cases, tracks are expected to appear infrequently, and false positives or missed detections should be penalized.

In Chapter V, we formalize and extend the commonly used priors in Bayesian tracking, originally introduced by Reid [130].

The likelihood can typically be factorized assuming conditional independence between targets, given the associations:

$$\begin{aligned}\mathbb{P}(Z_{1:T} \mid \Gamma_{1:T}) &= \prod_{i=1}^{N} \mathbb{P}(Z^{\gamma^i_{1:T}}) \\ &= \prod_{i=1}^{N}\prod_{t=1}^{T} \mathbb{P}(Z_t^{\gamma^i_t} \mid Z^{\gamma^i_{1:t-1}})\end{aligned} \tag{II.91}$$

Under the 1-to-1 association constraint, the per-frame measurement likelihood becomes:

$$\mathbb{P}(Z_t^{\gamma^i_t} \mid Z^{\gamma^i_{1:t-1}}) = \begin{cases} 1 & \text{if } \gamma^i_t = \emptyset \\ p(\mathbf{z}^j_t \mid Z^{\gamma^i_{1:t-1}}) & \text{if } \gamma^i_t = \{j\} \\ 0 & \text{otherwise} \end{cases} \tag{II.92}$$

where $p(\mathbf{z}^j_t \mid Z^{\gamma^i_{1:t-1}})$ is the predicted measurement likelihood from a single target Bayesian filtering (see Equation (II.54)).

Assuming a decoupled prior of the form (see Chapter V):

$$\mathbb{P}(\Gamma_{1:T}) = \prod_{i=1}^{N}\prod_{t=1}^{T} g(\gamma^i_t \mid \gamma^i_{1:t-1}), \tag{II.93}$$

we can rewrite the MAP estimation over $\Gamma_{1:T}$ as a cost minimization:

$$\Gamma^{\star}_{1:T} \in \arg\min_{\Gamma_{1:T}} \sum_{i=1}^{N}\sum_{t=1}^{T} c(Z_t^{\gamma^i_t} \mid Z^{\gamma^i_{1:t-1}}) \tag{II.94}$$

where the per-step cost c is:

$$c(Z_t^{\gamma_t^i} \mid Z^{\gamma_{1:t-1}^i}) = -\log\left[g(\gamma_t^i \mid \gamma_{1:t-1}^i) \cdot \mathbb{P}(Z_t^{\gamma_t^i} \mid Z^{\gamma_{1:t-1}^i})\right] \tag{II.95}$$

This leads directly to the generic linking problem introduced in Chapter II.1. Any of the linking algorithms discussed there can be used to solve the association problem, although some (*e.g.*, graph-based methods) require additional assumptions (*e.g.*, Markovian costs assumption) to guarantee tractability. Note that MHT with full depth solves precisely this optimization problem.

State Estimation. Assuming conditional independence between targets given the assignments, the states posterior distribution factorizes:

$$\mathbb{P}(X_{1:T} \mid \Gamma_{1:T}, Z_{1:T}) = \prod_{i=1}^{N} \mathbb{P}(X_{1:T}^i \mid Z^{\gamma_{1:T}^i}) \tag{II.96}$$

The state estimation can then be resolved target by target as a single-target smoothing problem:

$$X_{1:T}^{\star,i} = \arg\max \mathbb{P}(X_{1:T}^i | Z^{\gamma_{1:T}^i}) \tag{II.97}$$

In the linear Gaussian case, the marginal distributions $\mathbb{P}(\mathbf{x}_t^i \mid Z^{\gamma_{1:T}^i})$ are Gaussian and can be estimated using the RTS smoother (see Chapter II.3.1.6). The joint distribution across time is therefore also Gaussian, and the means $(\hat{\mathbf{x}}_t^{\mathrm{s,i}})_{1:T}$ provides the MAP estimate for $X_{1:T}^i$.

II.3.3 Random Finite Sets (RFS)

The Random Finite Set (RFS) framework [169,170] provides a mathematically rigorous approach to multi-object tracking under uncertainty in both the number and states of targets. In this framework, the multi-target state at time t is represented as a finite set:

$$X_t = \{\mathbf{x}_t^1, \ldots, \mathbf{x}_t^{N_t}\} \in \mathcal{F}(\mathcal{X}), \tag{II.98}$$

where $\mathcal{X} \subset \mathbb{R}^{d_\mathbf{x}}$ is the single-target state space, and $\mathcal{F}(\mathcal{X})$ denotes the collection of all finite subsets of $\mathcal{X}$. The cardinality $N_t = |X_t|$ is itself a random variable, reflecting an unknown number of targets.

Similarly, the measurement set at time t is modeled as a finite set:

$$Z_t = \{\mathbf{z}_t^1, \ldots, \mathbf{z}_t^{M_t}\} \in \mathcal{F}(\mathcal{Z}), \tag{II.99}$$

with $\mathcal{Z} \subset \mathbb{R}^{d_\mathbf{z}}$ the measurement space and M_t the random number of detections at time t.

This contrasts with the formalism introduced in Chapter II.1, where all N targets are defined over the full temporal interval $t \in [\![1, T]\!]$ using dummy states before birth or after death. In the RFS setting, only active targets are represented in X_t, allowing target births and deaths to be handled implicitly as changes in set cardinality.

RFS-based tracking enables unified modeling of target birth, death, detection uncertainty, and clutter without requiring explicit data association. While exact Bayesian inference over $\mathcal{F}(\mathcal{X})$ is generally intractable, several approximations allow practical implementations. The most widely known are the Probability Hypothesis Density (PHD) and Generalized Labeled Multi-Bernoulli (GLMB) filters.

II.3.3.1 Probability Hypothesis Density (PHD)

The PHD filter [171] approximates the multi-object posterior by its first-order moment — the intensity function $v_t(\mathbf{x})$, which is a function over the state space $\mathcal{X}$. It satisfies:

$$\int_S v_t(\mathbf{x})\,\mathrm{d}\mathbf{x} = \text{Expected number of targets in } S \subset \mathcal{X}. \tag{II.100}$$

In practice, peaks of $v_t(\mathbf{x})$ correspond to likely target positions.

The PHD is propagated recursively using a prediction-update cycle, analogous to Bayesian filtering:

$$\bar{v}_t(\mathbf{x}_t) = \gamma_t(\mathbf{x}_t) + \int p_S(\mathbf{x}_{t-1}) f(\mathbf{x}_t \mid \mathbf{x}_{t-1}) \hat{v}(\mathbf{x}_{t-1})\,\mathrm{d}\mathbf{x}_{t-1} \qquad \text{[prediction]} \tag{II.101}$$

$$\hat{v}_t(\mathbf{x}_t) = [1 - p_D(\mathbf{x}_t)]\bar{v}_t(\mathbf{x}_t) + \sum_{\mathbf{z}\in Z_t} \frac{p_D(\mathbf{x}_t) h(\mathbf{z} \mid \mathbf{x}_t) \bar{v}_t(\mathbf{x}_t)}{\kappa(\mathbf{z}) + \int p_D(\mathbf{x}') h(\mathbf{z} \mid \mathbf{x}') \bar{v}_t(\mathbf{x}')\,\mathrm{d}\mathbf{x}'} \qquad \text{[update]} \tag{II.102}$$

where γ_t is the birth intensity, p_S the survival probability, p_D the detection probability, f the dynamic model, h the measurement model, and κ the clutter intensity (false positives).

As the PHD filter integrates on the state-space $\mathcal{X}$ rather than on the much larger RFS state space $\mathcal{F}(\mathcal{X})$, it can be approximately integrated. A common close-form solution has been developed for Gaussian and linear processes (as for Kalman filters): the GM-PHD [172].

Limitations. While the PHD filter is efficient and elegant, it has two important limitations:

- **Moment approximation:** The PHD propagates only the first moment of the multi-object posterior, and estimating the states requires additional heuristics, like clustering. In GM-PHD, an ad-hoc procedure generates a state from the mean of each Gaussian component with sufficiently large weight (> 0.5).
- **No target identity:** The PHD is a function over $\mathcal{X}$ that does not maintain labels or identities. This limits its use in applications where consistent identities across time is essential. For example, in Baxter [119], Viterbi Track Linking (see Chapter II.1.3.2) is used as a post-processing of GM-PHD to recover target identities.

II.3.3.2 Generalized Labeled Multi-Bernoulli (GLMB)

The GLMB filter [173, 174] overcomes the limitations of PHD by augmenting each target state with a unique label: $(\mathbf{x}_t^i, \ell^i) \in \mathcal{X} \times \mathcal{L}$, with $\mathbf{L} \subset \mathbb{N}$. A labeled RFS is defined as a finite subset of distinctly labeled states:

$$\mathbf{X}_t = \{(\mathbf{x}_t^i, \ell^i)\}_{i=1}^{N_t}, \quad \ell^i \neq \ell^j \text{ for } i \neq j \tag{II.103}$$

The multi-object posterior is modeled using the GLMB family of distributions:

$$\pi_t(\mathbf{X}) = \Delta(\mathbf{X}) \sum_{\xi} w^{(\xi)}(L(\mathbf{X})) \prod_{(\mathbf{x},\ell)\in\mathbf{X}} p^{(\xi)}(\mathbf{x}, \ell), \tag{II.104}$$

Here, $\Delta(\mathbf{X})$ ensures label distinctness, ξ indexes data association hypotheses, with $w^{(\xi)}$ its weight and $p^{(\xi)}$ the corresponding single-target density.

Crucially, the GLMB family is closed under the Bayesian filtering recursion (prediction and update), enabling tractable implementation. Low-weight components can be pruned, and each posterior component corresponds to a distinct hypothesis on the data association and object identities.

Discussion. GLMB filtering offers a fully Bayesian multi-object tracking solution with the following advantages:

- **Identity preservation:** Track labels are maintained explicitly through time, ensuring consistent identities without post-hoc linking.
- **Probabilistic associations.** Ambiguity in measurement-to-track association is marginalized, whereas MHT takes binary decisions.
- **Bayesian modeling of events:** Birth, death, missed and false positive detections are all modeled through the RFS framework.

However, GLMB also presents practical challenges:

- **Computational cost:** Although hypothesis pruning maintains tractability, GLMB remains more computationally expensive than simpler approaches.
- **No smoothing:** The GLMB filter estimates the filtering distribution $p(\mathbf{X}_t \mid Z_{1:t})$. In contrast, MHT is a fixed lag smoother on the latent association variable, that maximizes $p(\Gamma_{1:t} \mid Z_{1:t+\tau_{\text{depth}}})$, potentially yielding more temporally consistent tracks. Look-ahead strategies, such as PHD look-ahead [174], attempt to address this limitation.
- **State extraction:** The filtering posterior is a mixture over numerous components; extracting a single realization $\mathbf{X}_t$ (*e.g.*, MAP or MMSE estimate) requires non-trivial selection heuristics. A typical one is to compute the MAP estimator of the cardinal N_t, and then select the N_t components with highest weights [174].

In summary, GLMB tracking offers a robust and theoretically sound alternative to MHT, particularly in settings with high false positives or sporadic measurements. Despite its theoretical appeal, GLMB remains less widely used in practice than simpler tracking-by-detection or MHT approaches, due to its complexity and computational cost.

II.4 MOT for Hydra's Neurons

In most biological tracking problems, the number of targets is both unknown and varies over time, due to targets entering or leaving the field of view. In our Hydra case, the number of neurons is fixed and unknown, but because of occlusions, out-of-focus, or their intermittent visibility, neurons may disappear for long time-range, leading, in practice, to a non-constant number of targets. This motivated our decision to adopt a tracking-by-detection framework for all algorithms developed in this thesis, as it naturally handles a variable number of targets. Moreover, this framework can incorporate any instance segmentation or detection method and thus benefits directly from recent—and future—advances in detection.

Although tracking performance is closely linked to detection accuracy in this paradigm, we found that detection methods were already reasonably effective in our context. In particular, deep learning models could be trained on relatively small annotated datasets (a few labeled patches). Consequently, we focused our efforts on improving the *linking* stage, which remains more challenging, in particular, due to the lack of annotated tracking data.

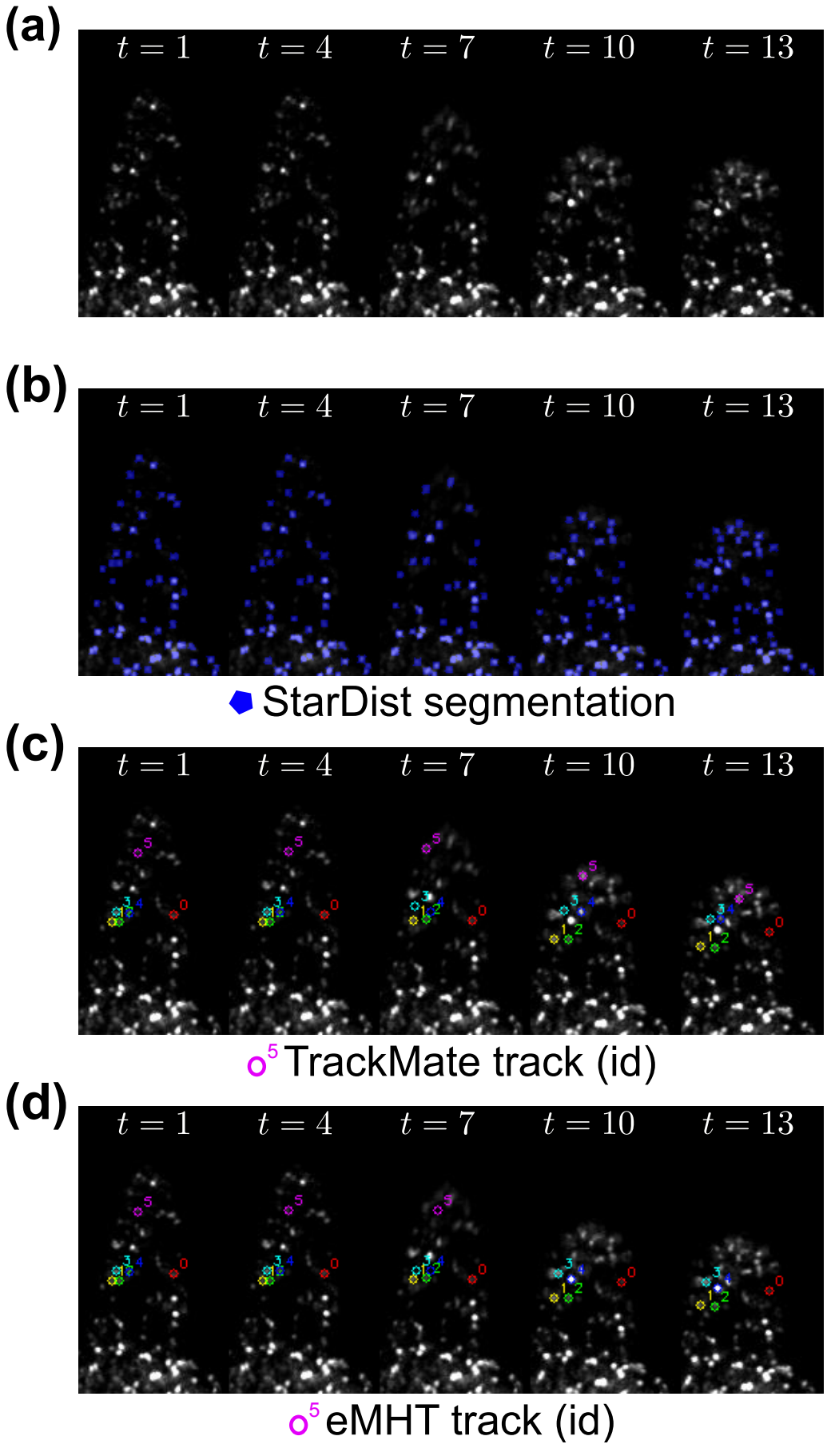

(a)
$t = 1$ $t = 4$ $t = 7$ $t = 10$ $t = 13$
(b)
$t = 1$ $t = 4$ $t = 7$ $t = 10$ $t = 13$
StarDist segmentation
(c)
$t = 1$ $t = 4$ $t = 7$ $t = 10$ $t = 13$
o^5 TrackMate track (id)
(d)
$t = 1$ $t = 4$ $t = 7$ $t = 10$ $t = 13$
o^5 eMHT track (id)

Figure II.8: Illustration of tracking errors from standard MOT approaches on a contracting tentacle. (a) TdTomato time-lapse sequence of a Hydra, zoomed in on a tentacle of interest. **(b)** StarDist detections [25] (in blue). Despite being trained on similar data, the model exhibits some missed and false positive detections, with an approximate F1 score of 90%. **(c)** Six selected tracks obtained with TrackMate [27,128]. Tracks 3, 4, and 5 exhibit several identity switches during the contraction; Track 1 (resp. 2) switches with the neuron originally tracked by 3 (resp. 4). Track 0 exhibits a single switch between frames 7 and 10, ending up on a different neuron. **(d)** The same six tracks obtained with eMHT [26]. Despite using a more advanced association mechanism, it suffers from similar tracking errors. Track 5 terminates early (frame 7), avoiding further errors, while Tracks 3 and 4 both switch identity between frames 4 and 7.

Hydra time-lapse sequences (e.g., Figure I.8) contain up to 1000 neurons tracked over 10,000 frames. This scale typically rules out computationally intensive graph-based methods (Chapter II.1.3.1). Instead, we relied on recursive approaches, using either LAP-based association strategies (Chapter II.1.3.3) or more sophisticated Multiple Hypothesis Tracking (MHT) (Chapter II.1.3.4). Due to the absence of ground-truth tracking annotations, we avoided supervised deep cost learning (e.g., Caliban [126], HCRM [133]) and focused exclusively on unsupervised or self-supervised techniques. In particular, we used Bayesian motion models that can be applied in recursive linking algorithms (see Chapter II.3.2).

Nevertheless, the specific challenges of neuron tracking in Hydra (outlined in Chapter I.4) render most standard linking algorithms ineffective. Figure II.8 illustrates these limitations using a time-lapse sequence focused on a contracting tentacle. The segmentation shown in Figure II.8.b was obtained using a trained StarDist model, which performs well at separating nearby neurons. However, neurons with low signal-to-noise ratio (SNR) often confuse the model, resulting in false positives and false negatives. These likely stem from inconsistencies in training annotations and limited inter-annotator agreement. On such sequences, StarDist typically achieves an F1 score of about 90%.

For linking, we qualitatively evaluate here two widely used algorithms:

- TrackMate [128], a Fiji-based [51] implementation of u-track [27], which uses a near-constant velocity Kalman filter and solves associations via a frame-by-frame LAP. It includes a global LAP-based post-processing step to handle missed detections.

- eMHT [26], implemented in Icy [50], which employs an Interacting Multiple Model (IMM) framework with Brownian and near-constant velocity Kalman filters, solving associations via Multiple Hypothesis Tracking.

As shown in Figure II.8, both methods struggle during periods of large motion, such as tentacle contractions. The underlying motion models fail to anticipate sudden displacements. Combined with the high neuron density and occasional detection errors, this leads to incorrect associations that cascade into further tracking errors in subsequent frames. To facilitate visualization, we uploaded videos[2] of the first 10s of this tentacle example tracked with different tracking methods (including our own methods).

[2] `https://partage.imt.fr/index.php/s/7cYJXqB2Qwao3a6`

While a few tracking errors may be tolerable, here, with the number of frames to consider (>1000) and with the number of errors seen on this tentacle example, many tracks end up being corrupted, impacting downstream analyses or necessitating extensive manual corrections. In this thesis, we propose more robust linking algorithms designed to significantly reduce such tracking errors.

Chapter III

SINETRA: a Synthetic Dataset for Evaluating Neuron Tracking

As discuss in Chapter I.4, accurately tracking neuronal activity in behaving animals presents significant challenges due to complex motions and background noise. The lack of annotated datasets limits the evaluation and improvement of such tracking algorithms. To address this, we developed *SINETRA*, a versatile simulator that generates synthetic tracking data for targets on a deformable background, closely mimicking live animal recordings. This simulator produces annotated 2D and 3D videos that reflect the intricate movements seen in behaving animals like *Hydra Vulgaris*. In this Chapter, we present this simulator, then benchmark three state-of-the-art tracking algorithms highlighting the current limitations of these methods in challenging scenarios and paving the way for improved cell tracking techniques in dynamic biological systems.

This work has been published at the IEEE ISBI conference in 2025 as: Raphael Reme*, Alasdair Newson, Elsa Angelini, Jean-Christophe Olivo-Marin, and Thibault Lagache. *SINETRA: A Versatile Framework for Evaluating Single Neuron Tracking in Behaving Animals.* IEEE ISBI, 2025 [65].

III.1 Motivation

To objectively compare cell tracking algorithms, several software tools [55, 175] and annotated datasets [59, 61, 176] have been developed. However, these mainly address classical cell tracking challenges, focusing on randomly moving and dividing cells *in vitro*, which do not reflect the conditions of neuron tracking in behaving animals. Neurons do not divide and their motion is driven by the animal's body deformations. As a result, evaluating neuron tracking algorithms typically requires labor-intensive manual annotations of time-lapse sequences, which is impractical for large datasets, especially in three dimensions [22, 28, 153]. Alternatively, synthetic simulations are used for targeted augmentation of neuron spots and volume registration in *C. Elegans* [177, 178].

To facilitate the objective comparison and improvement of SIngle NEurons TRAcking algorithms, we developed *SINETRA*, a versatile framework for simulating two- and three-dimensional time-lapse sequences reproducing the complex motions of neurons in live fluorescence imaging of behaving animals. This simulator models targets (neurons) as spots within a structured background that accounts for the auto-fluorescence of the embedding tissue. The tissue deformations related to animal behaviors are either modeled with a system of damped harmonic

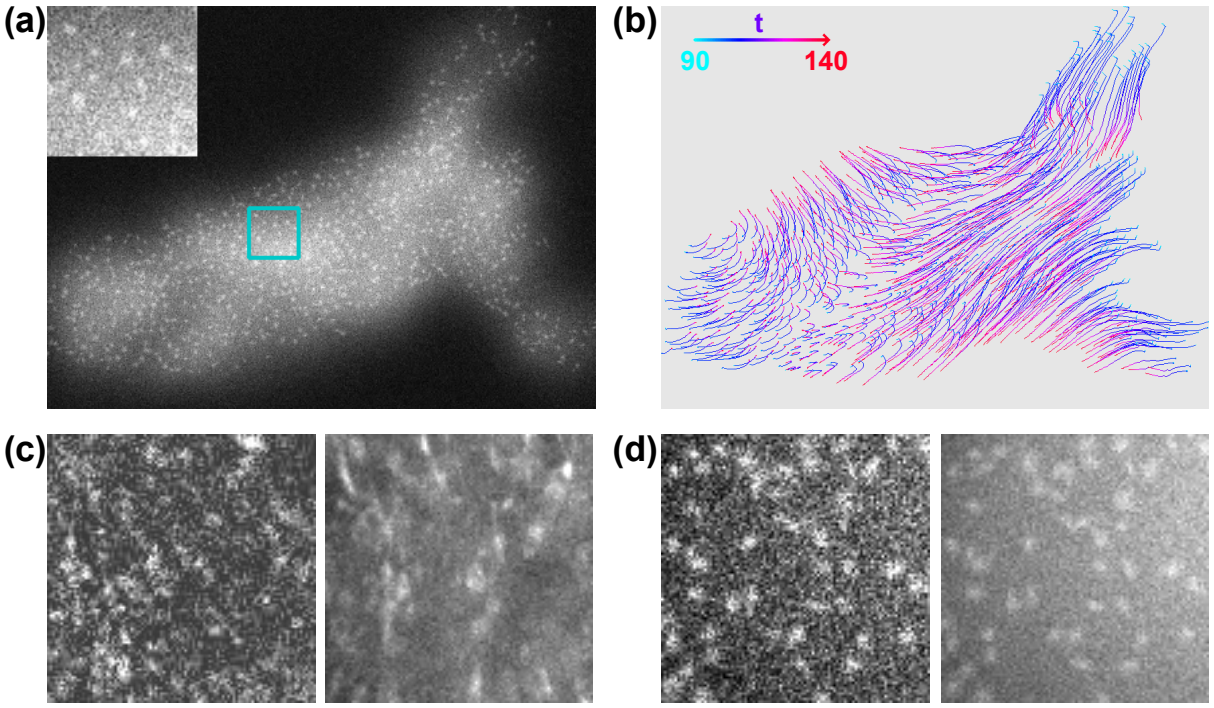


Figure III.1: Synthetic image simulator (Hydra flow) **(a)** A synthetic image using the default imaging parameters $\alpha = 0.2$ and $\Delta = 50$. **(b)** Synthetic tracks on 50 frames centered around a contraction (frames 90 to 140, with a contraction at $t = 100$). **(c)** 100x100 pixels from fluorescence videos of *Hydra Vulgaris*' neurons [10, 22]. **(d)** 100x100 pixels from synthetic videos using different imaging parameters. Left: $\Delta = 15$, $\alpha = 0.5$. Right: $\Delta = 200$, $\alpha = 0.2$.

oscillators subjected to random contraction or elongation forces, or are directly extracted from optical flow estimates of live animal deformations.

Using this simulation framework, we benchmarked three tracking algorithms on our synthetic data: *eMHT* [26] and *u-track* [27] that are standard Bayesian filtering and distance minimization methods, and *ZephIR* [153] which propagates semi-annotated tracks using image registration. This benchmark underscores the existing limitations of these approaches when dealing with complex movements, opening the door for enhanced cell tracking methods in dynamic biological systems (some developed in the following chapters of this thesis).

III.2 Method

The outputs of our simulator are a time-lapse sequence of images $(I_t)_{1\leq t\leq T}$ of size $\mathbf{S} \in \mathbb{N}^d$ pixels (with $d \in \{2, 3\}$ the dimension of images) and the tracks $(\mathbf{x}_t^i)_{1\leq t\leq T,\, 1\leq i\leq N}$, where T is the number of frames, N the number of tracks (see Figure III.1). At each time step t, $I_t \in [0,\, 1]^{\mathbf{S}}$ is the noisy fluorescence image of the targets at positions $\mathbf{x}_t \in \mathbb{R}^{N\times 2}$.

III.2.1 Modeling Fluorescent Targets and Background

III.2.1.1 Targets

We first modeled simple but realistic fluorescent targets (neurons). A target is characterized by its position, intensity and shape, all of which change over time. Formally, each target $1 \leq i \leq N$ in frame $1 \leq t \leq T$ is modeled with a $d \in \{2, 3\}$ dimensional Gaussian profile, with $\mathbf{x}_t^i \in \mathbb{R}^d$

its position, $w_t^i \in [0,\,1]$ its intensity weight, and $\mathbf{\Sigma}_t^i \in \mathcal{S}_d^{++}$ its elliptic shape, where $\mathcal{S}_d^{++}$ is the ensemble of positive-definite symmetric matrices in dimension d.

On the first frame, we defined the animal's body as the pixels $B \subset \mathbb{W}$, where $\mathbb{W} = \{\mathbf{w} \in \mathbb{N}^d, \mathbf{w} \geq 0 \text{ and } \mathbf{w} < \mathbf{S}\}$ is the simulation domain and $\mathbf{S} \in \mathbb{N}^d$ the dimensions of the generated images. These pixels B are either obtained from experimental videos using coarse segmentation of the tissue based on intensity thresholding or randomly sampled as an ellipse that covers around 30% of the surface/volume of the simulation domain $\mathbb{W}$. The initial positions of targets $\mathbf{x}_{t=0}^i$ are uniformly distributed within B with the sparsity of the distribution tuned via a minimal distance between targets (neurons). Target intensity weights $w_{t=0}^i$ are initially set to 1 and the axes of elliptic shapes $\mathbf{\Sigma}_{t=0,i}$ are initially sampled with a size from 1 to 3 pixels and randomly rotated (see Chapter III.2.1.4).

Let $\Theta_t^i = (w_t^i, \mathbf{x}_t^i, \mathbf{\Sigma}_t^i)$ be the parameters of a target i at frame t. The noise-free image of fluorescent targets I_t^{p} is defined at pixel location $\mathbf{w} \in \mathbb{W}$ as:

$$I_t^{\mathrm{p}}(\mathbf{w}) = \sum_{i=1}^{N} q(\mathbf{w};\ \Theta_t^i) = \sum_{i=1}^{N} w_t^i e^{-\frac{1}{2}(\mathbf{x}_t^i - \mathbf{w})^{\mathrm{T}} \mathbf{\Sigma}_t^{i\,-1} (\mathbf{x}_t^i - \mathbf{w})} \tag{III.1}$$

with $q(\mathbf{w};\ \Theta_t^i)$ the weighted Gaussian probability density function evaluated at pixel $\mathbf{w}$.

Since the intensity weights w_t^i are constrained in $[0,\,1]$ and the targets are separated by a minimal distance, the brightest pixels in I_t^p will not exceed the maximum gray-level value $(\forall \mathbf{w}, I_t^p(\mathbf{w}) < 1)$.

III.2.1.2 Background & Noise

We model background fluorescence as multiple Gaussian profiles of parameters $(\Theta_t^{i,\mathrm{b}})_{1 \leq i \leq N^{\mathrm{b}}} = (w_t^{i,\mathrm{b}}, \mathbf{x}_t^{i,\mathrm{b}}, \mathbf{\Sigma}_t^{i,\mathrm{b}})_{1 \leq i \leq N^{\mathrm{b}}}$. Positions of the background Gaussian profiles are also initially distributed homogeneously in B. The sizes of background profiles are larger than those of the fluorescent targets, ranging from 20 to 60 pixels. The noise-free background image $I^{\mathrm{b}}(\mathbf{w})$ is defined at pixel location $\mathbf{w}$ as:

$$I_t^{\mathrm{b}}(\mathbf{w}) = \sum_{i=1}^{N^{\mathrm{b}}} q(\mathbf{z};\ \Theta_t^{i,\mathrm{b}}) \tag{III.2}$$

The background profiles overlap extensively, leading to sums of intensities greater than the weights $w_t^{i,\mathrm{b}}$. Without scaling, the brightest pixels of I_t^{b} would exceed the maximum gray-level value. Therefore, we normalize the background images by a fixed constant, which is defined as the maximum value of the *first* background image: $G^{\mathrm{b}} = \max_{\mathbf{w}} I_0^{\mathrm{b}}(\mathbf{w})$.

Target and background signals are linearly mixed with a proportion α. We model the fluctuation of the number of photons detected at each pixel $\mathbf{w} \in \mathbb{W}$ using a Poisson Shot Noise process with Δ the integration time. The noisy image at frame t is generated as:

$$\begin{aligned} \bar{I}_t(\mathbf{w}) &= \alpha I_t^{\mathrm{p}}(\mathbf{w}) + \frac{(1-\alpha)}{G^{\mathrm{b}}} I_t^{\mathrm{b}}(\mathbf{w}) \\ I_t(\mathbf{w}) &\sim \frac{1}{\Delta} \mathcal{P}\left(\Delta \bar{I}_t(\mathbf{w})\right) \end{aligned} \tag{III.3}$$

This allows to parametrize the simulator to produce faithful images: α controls the visibility of targets over the background, and Δ the magnitude of the Poisson shot noise (see Figure III.1). By default, we use $\alpha = 0.20$ and $\Delta = 50$ (Figure III.1.a).

III.2.1.3 Intensity Variations

Our simulation framework allows to model the dynamics of target w_t^i intensity. This is particularly useful to account for changes in target fluorescence due to the biophysics of the dye, such as calcium indicators in fluorescence imaging of neural activity [15, 19]. Nonetheless, for the sake of simplicity, we have fixed the intensity weights to a constant ($\forall i, j, t,\ w_{t,i} = w^{\mathrm{b}}_{t,j} = 1$) in all our experiments.

III.2.1.4 Shape Evolution

For each Gaussian profile (target and background), the covariance matrix of the profile can be separated into size and rotation components:

$$\mathbf{\Sigma}_t^i = \mathbf{R}_{\boldsymbol{\theta}_t^i}^{\mathsf{T}} \mathrm{Diag}(\boldsymbol{\sigma}_t^i)^2 \mathbf{R}_{\boldsymbol{\theta}_t^i} \tag{III.4}$$

where $\boldsymbol{\sigma}_t^i \in \mathbb{R}_+^d$ are the sizes along each axes of the ellipse, and $\mathbf{R}_{\boldsymbol{\theta}_t^i}$ is the rotation matrix of angle $\boldsymbol{\theta}_t^i$. In the 2D case, $\boldsymbol{\theta}_t^i \in \mathbb{R}$ is a single angle of rotation around the z-axis. In 3D, $\boldsymbol{\theta}_t^i \in R^3$ are the rotation angles around the x, y and z axes.

We model the smooth random evolution of profiles over time around an equilibrium position with damped harmonic oscillators with random forces (see Chapter III.2.2.3). The equilibrium value of the rotation angle of a profile i is $\boldsymbol{\theta}^i_{\mathrm{eq}} = \boldsymbol{\theta}^i_{t=0} \sim \mathcal{U}([0, \pi])$. Then, to simulate rotation noise, we suppose that random forces are applied to the targets, modeled as i.i.d. Gaussian noise. We choose the magnitude of the force so that $\boldsymbol{\theta}_t^i \sim \mathcal{N}\left(\theta^i_{\mathrm{eq}}, \frac{\pi}{30}\right)$. We emphasize that $(\theta_t^i)_{0 \leq t \leq T}$ are correlated in time and are computed by solving the damped harmonic oscillators equations (and not sampled from this Gaussian distribution).

The initial profile sizes $\boldsymbol{\sigma}^i_{t=0}$ are sampled from a Uniform distribution (1 to 3 pixels for targets, 20 to 60 pixels for background profiles). Intuitively, larger targets should have larger size variations. We therefore define the normalized size as $\mathbf{s}_t^i = \frac{\boldsymbol{\sigma}_t^i}{\boldsymbol{\sigma}_{t=0}^i} \in R_+^d$. Random forces are applied to increase or decrease the normalized size around its equilibrium value of 1. They are modeled as i.i.d. Gaussian noise and chosen so that the $\mathbf{s}_t^i \sim \mathcal{N}(1, 0.05)$.

For both angle and size oscillators, we set the critical time of the system to $\tau = 10$ frames, and choose the dampening coefficient so that the system is critically damped (See Chapter III.2.2.3).

III.2.2 Motion Modeling

To model the motion of neurons embedded in a deformable tissue, we propose two elastic motion models. At each time-step t, we update the position of both targets $\mathbf{x}_t$ (neurons) and background profiles $\mathbf{x}_t^{\mathrm{b}}$ using the computed tissue deformation.

III.2.2.1 Optical Flow Motion

We estimate the animal deformation by computing optical flows between consecutive frames of an experimental video. The optical flows are computed with Farneback algorithm [179], an analytical algorithm that proved to be fast and robust in many applications. The positions of neurons and background profiles are initially sampled within B that corresponds to the thresholded mask of the behaving animals' body, then they are iteratively moved from the calculated flows at each time-step (see Figure III.1).

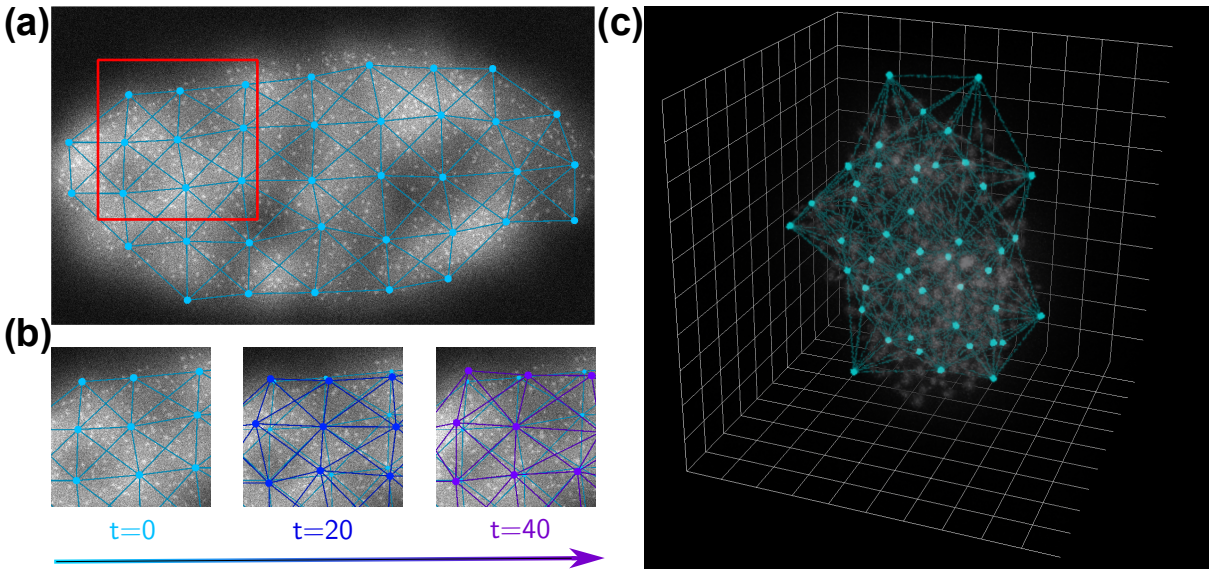


Figure III.2: Synthetic image simulator (Springs 2D/3D). (a): Visual appearance of the initial 2D frame with springs between control points (in blue). **(b)**: Motion induced by random forces and springs constraints over time. The motion of each particle (bright spots) is determined by interpolating control points. **(c)**: 3D projection of springs (in blue) and noisy targets (in white).

III.2.2.2 Springs Motion

To simulate random localized deformations of the animal's body, we employ a system of damped springs subjected to random contraction and elongation forces. We position a grid of n control points of equal mass attached by springs within the initial domain B (see Figure III.2). We then apply random forces to the control points and solve the n-body damped harmonic oscillator equations to estimate the dynamics of control points over time (see Chapter III.2.2.3).

Please note that the control points are not the targets and are not visible on the generated images. The positions of the targets and background Gaussian profiles are computed with an elastic interpolation (Thin Plate Spline) between the n control points.

More formally, let $(\mathbf{p}_i(t) \in \mathbb{R}^d)_{1 \leq i \leq n}$ be the coordinates of the n control points of our system at time t. For each point $\mathbf{p}_i$, we define its dampening coefficient $\lambda_i \in \mathbb{R}_+$. For each pair $(\mathbf{p}_i, \mathbf{p}_j)$, a spring is created with a stiffness $k_{ij} \in \mathbb{R}_+$ and an equilibrium length $l_{ij}^{\text{eq}} \in \mathbb{R}_+$. We set $k_{ij} = 0$ (no spring) for all pairs except the 8 (26 in 3D) closest neighbors. Let $l_{ij}(t)$ be the Euclidean distance between $\mathbf{p}_i$ and $\mathbf{p}_j$ at time t. The n-body damped harmonic oscillators can be written:

$$\ddot{\mathbf{p}}_i(t) = \mathbf{f}_i^{\text{springs}}(t) - \lambda_i \dot{\mathbf{p}}_i(t) + \mathbf{f}_i^{\text{rand}}(t) \tag{III.5}$$

$$\mathbf{f}_i^{\text{springs}}(t) = -\sum_j k_{ij} \left(l_{ij}(t) - l_{ij}^{\text{eq}} \right) \frac{\mathbf{p}_i(t) - \mathbf{p}_j(t)}{l_{ij}(t)} \tag{III.6}$$

where $\mathbf{f}_i^{\text{rand}}(t)$ is a random contraction or elongation force applied on control point i at time t. To compute this force, we start by randomly selecting a subset $S(t) \subset [\![1, n]\!]$ of control points, where $|S(t)| \sim U([\![2, m]\!])$ and $m = 10$ is the fixed maximum number of control points involved in the force. Then, we apply a contraction (resp. elongation) force toward (resp. from) the barycenter of the selected control points. Formally, let $\bar{\mathbf{p}}(t) = \frac{1}{|S(t)|} \sum_{i \in S(t)} \mathbf{p}_i(t)$ be the barycenter of the selected control points. Then:

$$\mathbf{f}_i^{\text{rand}}(t) = \begin{cases} 0 & \text{if } i \notin S(t) \\ d(t)a_i(t)\dfrac{\mathbf{p}_i(t) - \bar{\mathbf{p}}(t)}{||\mathbf{p}_i(t) - \bar{\mathbf{p}}(t)||_2} & \text{otherwise} \end{cases} \tag{III.7}$$

where $d(t) \in \{-1, 1\}$ controls the random direction of the motion (contraction or elongation) and $a_i(t) \sim U(\frac{1}{2}a_{\max}, a_{\max})$ is the random amplitude of the motion.

III.2.2.3 Damped Harmonic Oscillators

In our simulator, we rely on damped harmonic oscillators to model the temporal evolution of positions, angles and sizes of targets. We detail here the calculations underlying these equations in the one-dimensional case with a single spring, which benefits from theoretical background. For our multiple interacting springs (Chapter III.2.2.2), we have extrapolated the results from this simpler case.

Let $x(t) \in R$ be the quantity of interest at time t, we have the following system:

$$\ddot{x}(t) = f(t) - \lambda\dot{x}(t) - k\left(x(t) - x_{eq}\right) \tag{III.8}$$

where $f(t)$ is a random force, λ the dampening coefficient, k the stiffness coefficient and x_{eq} the equilibrium state of the system. The mass of the system is canceled out for simplicity.

The system is discretized with a time interval dt: $x_n = x(ndt)$. We decided to use the semi-implicit Euler method which is a symplectic integration method suited for our Hamiltonian system. Starting from initial conditions $x_0, \dot{x}_0$, it computes the next state following:

$$\begin{aligned} \ddot{x}_n &= f_n - \lambda\dot{x}_n - k(x_n - x_{eq}) \\ \dot{x}_{n+1} &= \dot{x}_n + dt\ddot{x}_n \\ x_{n+1} &= x_n + dt\dot{x}_{n+1} \end{aligned} \tag{III.9}$$

For the processes modeled here, oscillations are not realistic. Therefore we decided to critically damp the system so that the transient solution (without random forces) decays to the equilibrium state. This implies to set $\lambda = 2\sqrt{k}$.

Finally, we introduce the critical time $\tau = \frac{2}{\lambda}$ which corresponds to the exponential decay of the transient solution. We decided to use a reasonable value of $\tau = 10$ frames: without forces, the system returns to the equilibrium in around 10 frames. In all our springs, we have $\lambda = \frac{2}{\tau}$ and $k = \frac{1}{\tau^2}$.

III.3 Evaluation and Results

III.3.1 Evaluation

Using our simulator, we generated a synthetic dataset of fluorescent neurons in a behaving animal. We simulated 800 neurons over 200 frames, using $\Delta = 50$ and $\alpha = 0.2$. We focused on three different scenarios:

1. **Hydra Flow**: we generated 1024x1024 images using optical-flow based motion. The flows are extracted from 200 selected frames of an experimental video where the *Hydra Vulgaris* is contracting [10].

2. **Springs 2D**: we used springs-based motion (with the amplitude of random forces $a_{\max} = 4$ pixels) to generate 1024x1024 images.

3. **Springs 3D**: we simulated 200x200x200 volumes with springs-based motion (with $a_{\max} = 3$ pixels).

To measure the accuracy of predicted tracks, we use the Higher Order Tracking Accuracy (HOTA) metric [180], which accounts for localization, detection, and association performance. The HOTA metric aligns well with human visual assessment of tracking accuracy and has been widely adopted in evaluating the performance of tracking algorithms for natural images [180].

The HOTA score is defined as the geometric mean of the detection score and the association score:

$$\text{HOTA} = \sqrt{\text{DetA} \cdot \text{AssA}} \tag{III.10}$$

where DetA is the Jaccard index for detection (i.e., the ratio of correct detections to the total number of correct, missed, and false detections), and AssA is the Jaccard index for the association task [180]. We use a matching tolerance of r pixels, meaning that a predicted track is considered correctly associated with a ground-truth target only if its position lies within r pixels of the ground truth. In practice, we set $r = 2.0$ pixels, corresponding to the typical size of the targets. Tracks deviating beyond this distance are considered biologically erroneous as they may compromise subsequent analyses.

For each tracking algorithm, a grid search has been performed to tune to a few key parameters, while keeping others to their default values. For each experiment, five random videos are generated using the same simulation parameters but different random seeds. We report the mean and standard deviation for tables or the 99.7% confidence interval for figures over these five random videos.

The simulator and the tracking experiments are available alongside this synthetic dataset at `https://github.com/raphaelreme/SINETRA`.

III.3.2 Evaluated algorithms

Using this synthetic dataset, we compared three state-of-the-art tracking algorithms that are based on distinct methodological frameworks: **(1)** U-TRACK [27] re-implemented in TRACKMATE [128] from Fiji software [51] is a global distance minimization algorithm in two steps (frame-to-frame linking, tracklet stitching). We used the advanced version, which models target motion with a near-constant velocity Kalman filter. **(2)** EMHT [26] from Icy software [50] is a probabilistic algorithm based on multiple motion models, with a probabilistic handling of tracks and a multiple hypothesis association method. With our density of targets, we were only able to run the multiple hypothesis association with a tree depth of $\tau_{\text{depth}} = 3$ frames. **(3)** ZEPHIR [153] is a frame-by-frame registration algorithm that propagates partially annotated tracks. We tested ZEPHIR with three (ZEPHIR@3) and ten (ZEPHIR@10) annotated frames.

Each algorithm requires a pre-detection of targets before solving the tracking. We chose to rely on the wavelet denoising and thresholding method described in [24] for which we measured a detection F1 score of around 85% across our synthetic dataset. This matches detection performance found for this method on experimental *Hydra vulgaris* data [22]. If deep learning algorithms like STARDIST [25] could be also be used (for which we measured around 93% F1 on our dataset), we decided to limit the analysis of this study to unsupervised approaches, both

Table III.1: HOTA at 2 pixels [180] of the different tracking algorithms on three synthetic scenarios. We report the mean and std on 5 different random simulations sharing the same parameters (except the random seed).

Scenario	Hydra Flow	Springs 2D	Springs 3D
u-track [27, 51, 128]	$67.7 \pm 5.7\%$	$80.3 \pm 3.7\%$	$70.8 \pm 6.2\%$
eMHT [26, 50]	$73.4 \pm 8.1\%$	$82.4 \pm 2.8\%$	$76.5 \pm 6.4\%$
ZephIR@3 [153]	$76.4 \pm 6.0\%$	$74.6 \pm 7.4\%$	$51.5 \pm 2.5\%$
ZephIR@10 [153]	$91.9 \pm 3.2\%$	$91.3 \pm 1.7\%$	$75.3 \pm 2.7\%$

for the detection and linking algorithm. The impact of the detections quality will be studied in the next Chapters.

III.3.3 Results

Table III.1 summarizes the performance of the different tracking algorithms on the three simulation scenarios. In this benchmark, Springs 3D scenario consistently seems harder than the Springs 2D scenario as it adds a third degree of freedom to the problem. The Hydra Flow scenario is also more challenging than Springs 2D, because of the larger and longer contraction that occurs, increasing linking difficulties for traditional algorithms.

u-track [27,51,128] models target motion using a near-constant velocity assumption. However, we observed that this assumption fails in the presence of sudden and large contractions typical of behaving animals like Hydra, resulting in poor tracking performance. eMHT, which incorporates multiple motion models and a Multiple Hypothesis Tracking framework, demonstrates greater robustness. Nevertheless, it still struggles to track rapidly moving neurons, particularly evident in the top-right corners of Figure III.3. On the Hydra Flow scenario, u-track achieves an average HOTA score of 67.7%, while eMHT performs slightly better, reaching 73.4%.

The registration-based method ZephIR [153] shows greater resilience to large animal deformations, leveraging carefully selected annotated reference frames. With just three annotated frames, representing elongated and contracted states of the synthetic animal, ZephIR outperforms both u-track and eMHT on the Hydra Flow scenario, achieving a HOTA of 76.4%. Yet, on springs-based videos where contractions are more frequent, ZephIR requires more annotated frames to outperforms u-track and eMHT. With 10 annotated frames on our dataset, it reaches above 90% HOTA on 2D scenarios. However, its performance degrades on 3D data, where registration precision declines, and it is outperformed by eMHT, which dot not require any annotation. Moreover, manual annotation of reference frames is time-consuming and limits ZephIR's applicability in highly deformable organisms and long-duration recordings. Additionally, ZephIR assumes a fixed number of targets based on the given annotated frames, reducing its capabilities for scenarios where objects enter or leave the field of view.

We also compared the computational efficiencies of the different tracking methods. u-track is the fastest method, running at approximately 25 frames per seconds (FPS) on both 2D and 3D sequences using a standard Lenovo Legion 5 laptop. eMHT, due to the complexity of its multiple hypothesis framework, operates at around 2 FPS on both 2D and 3D sequences. ZephIR is the slowest option, as it requires the optimization of its objective function on each

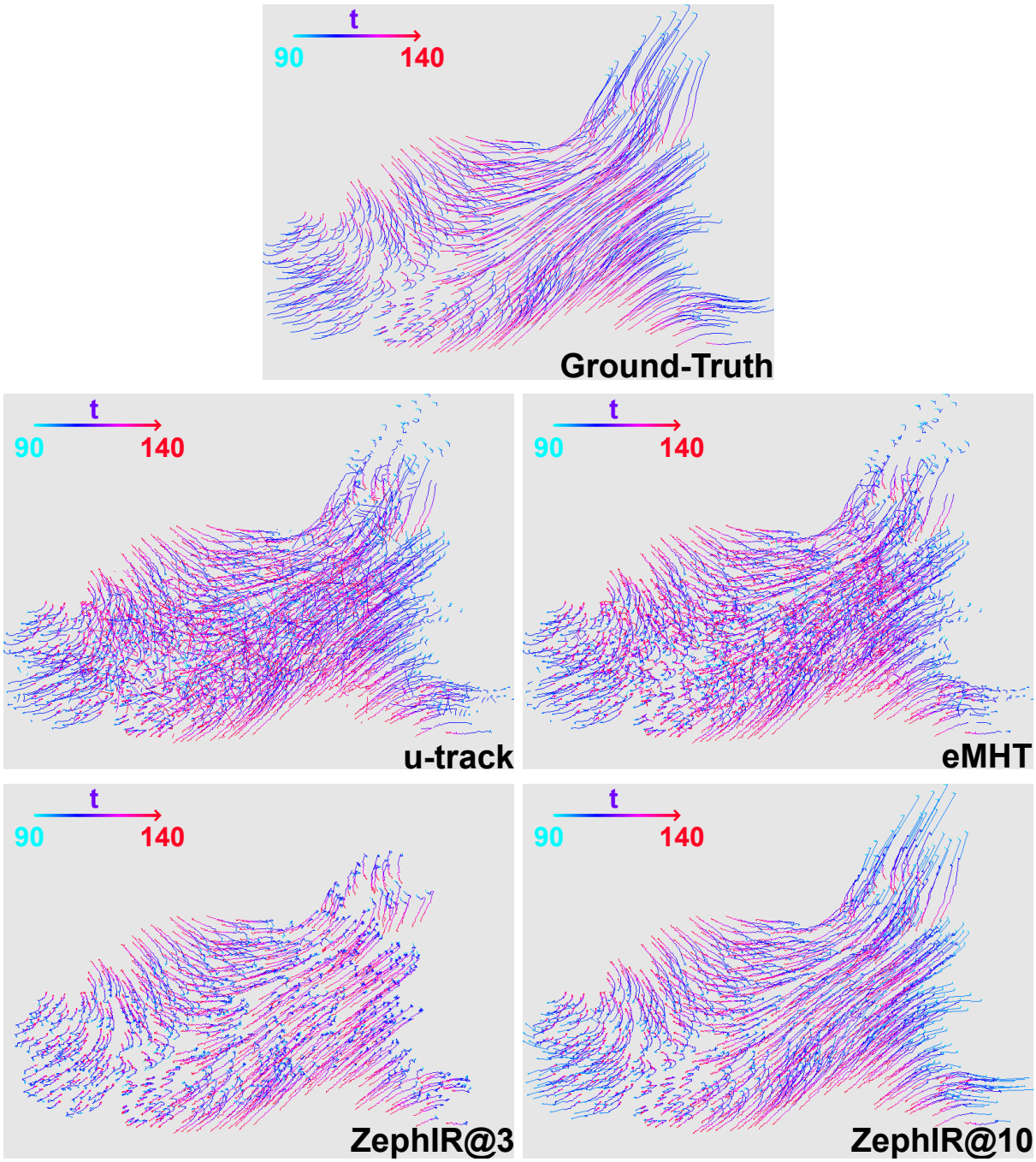


Figure III.3: Temporal projection of tracks (Hydra flow scenario) on 50 frames centered around a contraction (frames 90 to 140, contraction at $t = 100$).

frame. On CPU, it runs at approximately 0.3 FPS, but GPU acceleration improves performance significantly: it reaches 3 FPS on 2D sequences using an Nvidia RTX 3070, though performance is notably slower in 3D.

III.4 Discussion

In this chapter, we presented *SINETRA*, a versatile framework for generating synthetic, annotated multiple object tracking (MOT) datasets, specifically designed to replicate realistic imaging noise and complex motion patterns such as those observed in neurons of behaving animals like *Hydra vulgaris* or *C. elegans*.

This simulator enables the quantitative evaluation of MOT algorithms in conditions closely resembling those encountered in practice when tracking neurons in Hydra. While it is particularly tailored to the challenges posed by neurons tracking, we emphasize that general purposes biological MOT algorithms should also perform well on such datasets as it expressed some of the key challenges found in biological tracking. However, several key design choices should be considered when interpreting results:

1. **Target shape**: Each synthetic target is modeled as an ellipse whose shape fluctuates slowly over time around a random equilibrium shape. This introduces slight visual differences between targets, aiding in identity preservation. However, this representation remains simplistic. Real neurons often exhibit more complex structures and textures (see Chapter VI) that could be leveraged for improved tracking. Although more realistic textures might be generated with modern deep generative models, we deemed the effort required to integrate such models disproportionate to the benefit in the context of MOT benchmarking.

2. **Noise modeling**: As with any synthetic dataset, our simulation of noise cannot fully reproduce the complexity of real image acquisition. We incorporate both Poisson (shot) noise and a background noise component simulating light backscattering within the synthetic tissue. While simplified, these two sources of noise offer a reasonably faithful approximation of real fluorescence microscopy data and are sufficient to expose the limitations of existing MOT algorithms.

3. **Motion**: We proposed two complementary motion generation methods. The first uses optical flow extracted from real videos of contracting Hydra to drive synthetic target trajectories. This produces highly realistic motion with minimal parameter tuning, assuming the quality of the optical flow is sufficient. However, this method may introduce bias, as the synthetic motion is inherently constrained by what optical flow can capture. To address this, we also implemented a fully synthetic motion model based on damped springs subject to stochastic contraction and elongation forces. This model is highly configurable, allowing the user to adjust difficulty by tuning parameters such as force magnitude and frequency. The default configuration mimics the acceleration profile of the Hydra Flow scenario and can be easily extended to 3D, enabling benchmarking of 3D MOT algorithms. In all SINETRA datasets, target motion are highly spatially correlated, as in experimental videos of neurons in behaving animals, we did not propose classical independent motion patterns in our scenarios.

4. **Temporal intensity dynamics**: While our simulator supports temporal evolution of target intensities through per-frame weights w_t, all current benchmarks in this chapter use a fixed value ($\forall t,\ w_t = 1.0$). This is realistic for fluorescence labels such as TdTomato

(see Figure I.8), whose signal is independent of neuronal activity. In future work, the simulator could be extended to model calcium dynamics for GCaMP-type indicators, therefore enabling the evaluation of tracklet stitching strategies discussed in Chapter VI.

Using this simulation framework, we performed a series of benchmark experiments in both 2D and 3D, evaluating three representative tracking methods. These experiments confirmed our earlier qualitative analysis (Chapter II.4), which suggested that classical MOT algorithms struggle to cope with targets undergoing sudden and large accelerations. Among the methods tested, ZephIR, which was specifically designed for tracking neurons in deformable organisms, demonstrated the best robustness to such motion. However, it lacks the flexibility of tracking-by-detection frameworks, as it assumes a fixed number of targets and requires manual annotation of key frames. This manual annotation is labor-intensive, error-prone, and requires domain expertise—factors that limit ZephIR's scalability for large datasets or long recordings.

Based on this analysis, the subsequent chapters of this thesis will introduce more robust MOT algorithms that significantly improve both quantitative and qualitative tracking performance—on both SINETRA-generated data and real experimental recordings of *Hydra vulgaris*.

Chapter IV

KOFT: Exploiting Optical Flow in Kalman-Based Tracking

As shown in previous chapters, standard tracking-by-detection MOT algorithms struggles when applied on our Hydra data or our SINETRA synthetic datasets. In this chapter, we propose a first contribution toward more robust MOT algorithms. By construction, standard Bayesian filtering, the foundation of most state-of-the-art non-supervised recursive linking algorithms, is unable to predict sudden changes in object velocity. To address this, we design a novel tracking approach that introduces optical flow as a second independent sensor to measure the object future velocity in a Kalman filtering framework. We evaluate our approach on the synthetic datasets presented in Chapter III and with annotated time-lapse calcium imaging of neuronal activity in *Hydra vulgaris*. Our results show that integrating optical flow estimations into Kalman filtering significantly enhances the accuracy of object position predictions, reducing tracking errors by more than twofold compared to other tracking algorithms.

This work has been published at the IEEE ISBI conference in 2024 as: Raphael Reme*, Alasdair Newson, Elsa Angelini, Jean-Christophe Olivo-Marin, and Thibault Lagache. *Particle tracking in biological images with optical-flow enhanced Kalman filtering.* IEEE ISBI, 2024 [66]. And it has been extended and submitted to IEEE TIP in 2025 as: Raphael Reme*, Alasdair Newson, Elsa Angelini, Jean-Christophe Olivo-Marin, and Thibault Lagache. *Using Optical Flow as a Velocity Sensor for Kalman-Based Tracking of Biological Objects.*

IV.1 Motivation

As detailed in Chapter I.4, this thesis focuses on non-supervised or self-supervised recursive linking algorithms within the tracking-by-detection framework (see Chapter II). A key component of most of these methods is Bayesian motion modeling, which predicts the next target position based on past associated detections [26, 57, 128, 132]. This principle was introduced in Chapter II.3.2: it allows the computation of the likelihood of the next measurement given the previous ones, $p(\mathbf{z}_t|\mathbf{z}_{1:t-1})$, where track indices are omitted for simplicity.

The effectiveness of Bayesian filtering in tracking critically depends on the choice of *a priori* models for both the process and measurement distributions (Eqs. II.51 and II.52). The measurement distribution is typically modeled as a Gaussian centered on the true object position, reflecting the localization uncertainty inherent in detection-based observations [26, 57, 124, 128, 132]. In contrast, the process model characterizes the expected dynamics of the tracked objects and determines the predictive power of the filter. A common approach for small biological objects is

to assume Brownian motion, where object displacements are random and the process distribution is modeled as a Gaussian centered on the current position [27]. Alternatively, near-constant velocity or acceleration models are often used, where velocity is estimated recursively from past positions [124, 128]. To accommodate multiple motion regimes (e.g., Brownian and directed), Interacting Multiple Models (IMM) can be employed, enabling the filter to switch between dynamic modes and capture transitions in motion behavior [26, 57, 132].

However, in all these Bayesian models, the detection process acts as the sole source of information (sensor), providing noisy measurements for each track restricted to the position of objects. Consequently, object motion is predicted exclusively from past potentially inaccurate positions ($\mathbf{z}_{1:t-1}$) and such models struggle to anticipate abrupt changes in object dynamics. In the case of Hydra, the animal's sudden behavior cannot be precisely predicted from past movements. As shown in Chapter I.4 and Chapter III, failing to anticipate these sudden motions often leads to incorrect associations, triggering cascades of tracking errors.

To overcome this limitation, we propose a novel approach that introduces optical flow [142] as an additional velocity sensor within a Kalman filtering framework. Critically, optical flow provides velocity measurements independent from detection-based positional measurements, allowing the Kalman filter to effectively fuse these complementary information to achieve more accurate state estimation. Unlike standard approaches that predict motion solely from associated detection from past frames, our method computes optical flow between the current and the future frame, providing measurements of target *future* velocity. This forward-looking prediction strongly improves robustness to sudden velocity changes. Moreover, as optical flow estimation imposes spatial regularity, this regularity is implicitly transferred to track velocities, which is particularly relevant in our Hydra case.

We design a Kalman filter based either on a Brownian motion or a near-constant velocity model, and perform two sequential updates at each time step t. The first update incorporates the position measurement obtained from our detection sensor. The second update integrates the independent velocity measurement derived from our optical flow sensor, computed by estimating the flow between frames t and $t+1$ at the predicted location of the track. This second update significantly improves velocity estimation, particularly under abrupt or irregular motion changes that standard Kalman filters—relying solely on past positions—fail to predict. This enhanced Kalman filtering approach is embedded within a simple yet effective online tracking algorithm that employs the linear assignment problem (LAP) to perform frame-by-frame association (see Chapter II.1.3.3). We refer to this method as Kalman and Optical Flow Tracking (KOFT).

Using both experimental and synthetic data, we begin with an objective comparison of the latest optical flow methods to identify the most suitable approach for our application. Next, we assess the tracking performance of KOFT on the challenging task of tracking fluorescent cells (neurons) in *Hydra vulgaris*, a highly deformable freshwater organism undergoing rapid body contractions and tentacle swaying [10,22]. Finally, leveraging the SINETRA synthetic simulator introduced in Chapter III, we evaluate the robustness of our algorithm across a variety of synthetic configurations (e.g., motion patterns, detection rates, fluorescence noise), demonstrating that KOFT outperforms other state-of-the-art tracking algorithms, halving tracking errors.

IV.2 Method

We introduce a Kalman filtering framework for multiple object tracking that combines two complementary sources of information: (1) position measurements obtained from object detections, and (2) velocity estimates derived from optical flow. The filter operates on a state representation

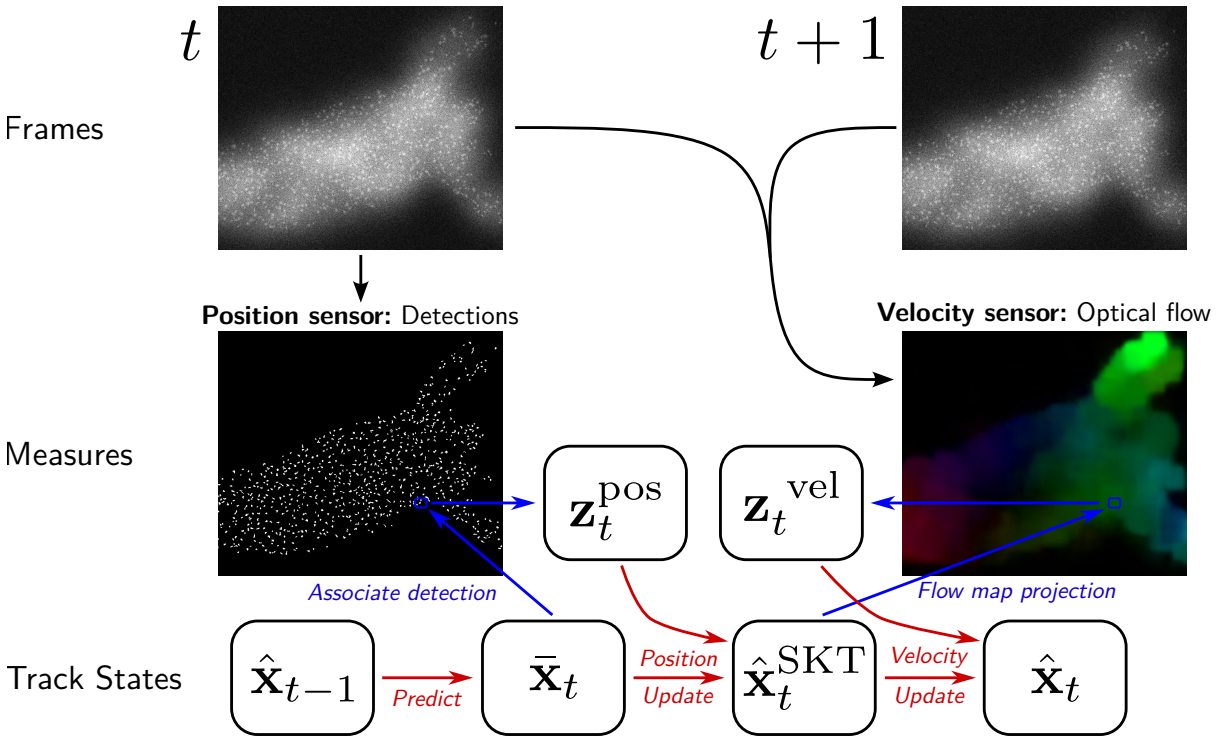


Figure IV.1: KOFT overview. We use a two-sensor Kalman filter to iteratively estimate object states (positions and velocities). First, the *a priori* estimate $\bar{\mathbf{x}}_t$ at frame t is predicted from the previous posterior estimate $\hat{\mathbf{x}}_{t-1}$ at frame $t-1$. In standard Kalman Tracking (SKT), the predicted track is associated with a detection from frame t, yielding a position measurement $\mathbf{z}_t^{\text{pos}}$ that is used to update the state to $\hat{\mathbf{x}}_t^{\text{SKT}}$. In our KOFT method, we compute a dense optical flow between frames t and $t+1$ to estimate the *future* displacement at each pixel. This independent sensor provides a velocity measurement $\mathbf{z}_t^{\text{vel}}$, which is integrated into the Kalman update to compute the posterior state $\hat{\mathbf{x}}_t$ at frame t. The resulting filter fuses both detection-based position measurements and optical flow-based velocity measurements to improve robustness and accuracy in dynamic tracking scenarios.

that explicitly includes the object's *future velocity*, enabling more accurate motion prediction. Our method employs a dual-update scheme to perform Bayesian-optimal fusion of the two independent measurements. This design naturally handles missing data from either sensor and improves tracking robustness in dynamic and noisy environments.
Building on this filtering framework, we introduce the *Kalman and Optical Flow Tracking (KOFT)* algorithm, along with a baseline variant, *Standard Kalman Tracking (SKT)*. SKT shares the same structure as KOFT but excludes the optical flow–based velocity updates, relying solely on detection-based position measurements.

IV.2.1 State-Space Modeling

Each track is modeled as a linear Gaussian Hidden Markov Model (HMM), defined by a hidden state vector $\mathbf{x}_t$ and two measurements at frame t: a position measurement $\mathbf{z}_t^{\text{pos}}$ from the detection process, and a velocity measurement $\mathbf{z}_t^{\text{vel}}$ from the optical flow algorithm. For clarity, we omit the track index i. We assume that tracks evolve independently, and that the process, position, and velocity measurement noises are mutually independent. The HMM is fully described

by the following equations:

$$\mathbf{x}_t \mid \mathbf{x}_{t-1} \sim \mathcal{N}(\mathbf{F}\mathbf{x}_{t-1}, \mathbf{Q}) \qquad \text{[process]} \tag{IV.1}$$

$$\mathbf{z}_t^{\text{pos}} \mid \mathbf{x}_t \sim \mathcal{N}(\mathbf{H}_{\text{pos}}\mathbf{x}_t, \mathbf{R}_{\text{pos}}) \qquad \text{[position measurement]} \tag{IV.2}$$

$$\mathbf{z}_t^{\text{vel}} \mid \mathbf{x}_t \sim \mathcal{N}(\mathbf{H}_{\text{vel}}\mathbf{x}_t, \mathbf{R}_{\text{vel}}) \qquad \text{[velocity measurement]} \tag{IV.3}$$

Here, $\mathbf{F}$ denotes the state transition matrix, $\mathbf{Q}$ the process noise covariance, $\mathbf{H}_{\text{pos}}$ and $\mathbf{H}_{\text{vel}}$ the measurement matrices for position and velocity, and $\mathbf{R}_{\text{pos}}$ and $\mathbf{R}_{\text{vel}}$ their respective noise covariances.

Although the two measurements could be combined into a single observation vector, our decoupled formulation offers greater flexibility—specifically for handling missing detections while still incorporating available optical flow measurements.

In this model, we assume that the positional and velocity measurements are independent. We verify this assumption empirically in Chapter IV.3.5.

IV.2.2 Kalman Filtering with Dual Updates

Under the assumptions of linear dynamics and Gaussian noise, Kalman filtering provides the Bayesian-optimal state estimates. At each time step t, we assume the previous filtering distribution is Gaussian:

$$\mathbf{x}_{t-1} \mid \mathbf{z}_{1:t-1}^{\text{pos}}, \mathbf{z}_{1:t-1}^{\text{vel}} \sim \mathcal{N}(\hat{\mathbf{x}}_{t-1}, \hat{\mathbf{P}}_{t-1}), \tag{IV.4}$$

where $\hat{\mathbf{x}}_{t-1}$ and $\hat{\mathbf{P}}_{t-1}$ denote the mean and covariance of the state estimate at time $t-1$.

The filtering distribution at time t is then computed via one prediction step followed by two sequential updates:

Prediction step. Using the process model, we compute the prior (predicted) distribution as

$$\bar{\mathbf{x}}_t = \mathbf{F}\hat{\mathbf{x}}_{t-1}, \tag{IV.5}$$

$$\bar{\mathbf{P}}_t = \mathbf{F}\hat{\mathbf{P}}_{t-1}\mathbf{F}^{\mathsf{T}} + \mathbf{Q}, \tag{IV.6}$$

where $\bar{\mathbf{x}}_t$ and $\bar{\mathbf{P}}_t$ represent the predicted state and its covariance.

First update (position sensor). If a detection $\mathbf{z}_t^{\text{pos}}$ is available, the predicted state is refined using the position measurement:

$$\mathbf{y}_t^{\text{pos}} = \mathbf{z}_t^{\text{pos}} - \mathbf{H}_{\text{pos}}\bar{\mathbf{x}}_t, \qquad \mathbf{S}_t^{\text{pos}} = \mathbf{H}_{\text{pos}}\bar{\mathbf{P}}_t\mathbf{H}_{\text{pos}}^{\mathsf{T}} + \mathbf{R}_{\text{pos}}, \tag{IV.7}$$

$$\hat{\mathbf{x}}_t^{\text{SKT}} = \bar{\mathbf{x}}_t + \mathbf{K}_t^{\text{pos}}\mathbf{y}_t^{\text{pos}}, \qquad \hat{\mathbf{P}}_t^{\text{SKT}} = (\mathbf{I} - \mathbf{K}_t^{\text{pos}}\mathbf{H}_{\text{pos}})\bar{\mathbf{P}}_t, \tag{IV.8}$$

where $\mathbf{y}_t^{\text{pos}}$ and $\mathbf{S}_t^{\text{pos}}$ denote the innovation and its covariance, and $\mathbf{K}_t^{\text{pos}} = \bar{\mathbf{P}}_t\mathbf{H}_{\text{pos}}^{\mathsf{T}}\mathbf{S}_t^{\text{pos}^{-1}}$ is the Kalman gain. If the detection is missing, the update is skipped and $\hat{\mathbf{x}}_t^{\text{SKT}} = \bar{\mathbf{x}}_t$.

Second update (velocity sensor). Assuming independence of measurements, we can write the update using the velocity measurement as:

$$p(\mathbf{z}_t^{\text{vel}} \mid \mathbf{z}_{1:t}^{\text{pos}}, \mathbf{z}_{1:t-1}^{\text{vel}}) = \int p(\mathbf{x}_t \mid \mathbf{z}_{1:t}^{\text{pos}}, \mathbf{z}_{1:t-1}^{\text{vel}}) p(\mathbf{z}_t^{\text{vel}} \mid \mathbf{x}_t) \mathrm{d}\mathbf{x}_t, \tag{IV.9}$$

$$p(\mathbf{x}_t \mid \mathbf{z}_{1:t}^{\text{pos}}, \mathbf{z}_{1:t}^{\text{vel}}) = \frac{p(\mathbf{x}_t \mid \mathbf{z}_{1:t}^{\text{pos}}, \mathbf{z}_{1:t-1}^{\text{vel}})\, p(\mathbf{z}_t^{\text{vel}} \mid \mathbf{x}_t)}{p(\mathbf{z}_t^{\text{vel}} \mid \mathbf{z}_{1:t}^{\text{pos}}, \mathbf{z}_{1:t-1}^{\text{vel}})}. \tag{IV.10}$$

Given the linear Gaussian model, this yields a second Kalman update:

$$\mathbf{y}_t^{\text{vel}} = \mathbf{z}_t^{\text{vel}} - \mathbf{H}_{\text{vel}}\hat{\mathbf{x}}_t^{\text{SKT}}, \qquad \mathbf{S}_t^{\text{vel}} = \mathbf{H}_{\text{vel}}\hat{\mathbf{P}}_t^{\text{SKT}}\mathbf{H}_{\text{vel}}^{\mathsf{T}} + \mathbf{R}_{\text{vel}}, \tag{IV.11}$$

$$\hat{\mathbf{x}}_t = \hat{\mathbf{x}}_t^{\text{SKT}} + \mathbf{K}_t^{\text{vel}}\mathbf{y}_t^{\text{vel}}, \qquad \hat{\mathbf{P}}_t = (\mathbf{I} - \mathbf{K}_t^{\text{vel}}\mathbf{H}_{\text{vel}})\hat{\mathbf{P}}_t^{\text{SKT}}, \tag{IV.12}$$

where $\mathbf{K}_t^{\text{vel}} = \hat{\mathbf{P}}_t^{\text{SKT}}\mathbf{H}_{\text{vel}}^{\mathsf{T}}\mathbf{S}_t^{\text{vel}^{-1}}$ is the Kalman gain for the velocity update. In the *Standard Kalman Tracking (SKT)* baseline, this step is omitted, and $\hat{\mathbf{x}}_t = \hat{\mathbf{x}}_t^{\text{SKT}}$.

Dual-update interpretation. From a Bayesian perspective, this two-step update is equivalent to a single Kalman update using a stacked observation model:

$$\mathbf{z}_t = \begin{pmatrix} \mathbf{z}_t^{\text{pos}} \\ \mathbf{z}_t^{\text{vel}} \end{pmatrix}, \quad \mathbf{H} = \begin{pmatrix} \mathbf{H}_{\text{pos}} \\ \mathbf{H}_{\text{vel}} \end{pmatrix}, \quad \mathbf{R} = \begin{pmatrix} \mathbf{R}_{\text{pos}} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{\text{vel}} \end{pmatrix}.$$

However, the sequential formulation provides greater flexibility by allowing partial updates when only one of the two sensors provides a measurement. For ablation purposes, we define KOFT--, a variant of our method that adopts the single-update formulation and does not exploit this flexibility.

IV.2.3 Measurement and Process Models

Let $\mathbf{w}_t \in \mathbb{R}^d$ denote the position of an object at time t. We define the hidden state vector as:

$$\mathbf{x}_t = \begin{pmatrix} \mathbf{w}_t \\ \mathsf{d}\mathbf{w}_t \end{pmatrix} \in \mathbb{R}^{2d}, \tag{IV.13}$$

where $\mathsf{d}\mathbf{w}_t = \mathbf{w}_{t+1} - \mathbf{w}_t$ represents the forward finite difference used as a proxy for *future* velocity.

The measurement models are straightforward: the detection sensor observes the current position, while the optical flow sensor provides an estimate of the forward displacement. The corresponding observation matrices and noise covariances are defined as:

$$\mathbf{H}_{\text{pos}} = \begin{pmatrix} \mathbf{I}_d & \mathbf{0}_d \end{pmatrix}, \qquad \mathbf{R}_{\text{pos}} = \sigma_{\text{pos}}^2\mathbf{I}_d, \tag{IV.14}$$

$$\mathbf{H}_{\text{vel}} = \begin{pmatrix} \mathbf{0}_d & \mathbf{I}_d \end{pmatrix}, \qquad \mathbf{R}_{\text{vel}} = \sigma_{\text{vel}}^2\mathbf{I}_d, \tag{IV.15}$$

where σ_{pos} and σ_{vel} quantify the uncertainty associated with the detection and optical flow sensors, respectively.

Two common motion models are used in biological object tracking: *Brownian motion* and *Directed motion.*

Brownian motion. In this model, the next position is modeled as a random walk:

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \boldsymbol{\epsilon}_t^{\text{mov}}, \quad \boldsymbol{\epsilon}_t^{\text{mov}} \sim \mathcal{N}(\mathbf{0}_d, \sigma_{\text{mov}}^2\mathbf{I}_d), \tag{IV.16}$$

where $\boldsymbol{\epsilon}_t^{\text{mov}}$ is i.i.d. Gaussian process noise. The corresponding state-space model is:

$$\mathbf{F}_{\text{brown}} = \begin{pmatrix} \mathbf{I}_d & \mathbf{I}_d \\ \mathbf{0}_d & \mathbf{0}_d \end{pmatrix}, \quad \mathbf{Q}_{\text{brown}} = \begin{pmatrix} \mathbf{0}_d & \mathbf{0}_d \\ \mathbf{0}_d & \sigma_{\text{mov}}^2\mathbf{I}_d \end{pmatrix}. \tag{IV.17}$$

This model assumes no velocity persistence (expected velocity is zero). The optical flow sensor thus provides a crucial correction to the predicted motion by injecting a forward-looking displacement estimate $\mathbf{z}_t^{\text{vel}}$.

Directed motion. For target with persistent velocity, we adopt a near-constant velocity model:

$$\mathsf{d}\mathbf{w}_{t+1} = \mathsf{d}\mathbf{w}_t + \boldsymbol{\epsilon}_t^{\text{mov}}, \quad \boldsymbol{\epsilon}_t^{\text{mov}} \sim \mathcal{N}(\mathbf{0}_d, \sigma_{\text{mov}}^2 \mathbf{I}_d), \tag{IV.18}$$

leading to the following transition and noise matrices:

$$\mathbf{F}_{\text{directed}} = \begin{pmatrix} \mathbf{I}_d & \mathbf{I}_d \\ \mathbf{0}_d & \mathbf{I}_d \end{pmatrix}, \quad \mathbf{Q}_{\text{directed}} = \begin{pmatrix} \mathbf{0}_d & \mathbf{0}_d \\ \mathbf{0}_d & \sigma_{\text{mov}}^2 \mathbf{I}_d \end{pmatrix}. \tag{IV.19}$$

This model favors smooth velocity evolution and benefits from the optical flow measurement, which balances previous displacement information with forward-looking velocity estimates, improving robustness to non-stationary dynamics.

IV.2.4 Track-to-Detection Association

For any track $\gamma_{1:t-1}^i$, Kalman filtering estimates the likelihood $p(\mathbf{z}_t^{\text{pos},j} \mid Z^{\gamma_{1:t-1}^i})$ of observing a candidate detection $\mathbf{z}_t^{\text{pos},j}$ at time t, given the track's previous associated measurements $Z^{\gamma_{1:t-1}^i}$. However, Kalman filtering alone does not perform the *linking* task itself.

In both SKT and KOFT, we adopt the online data association framework described in Chapter II.1.3.3, where a Linear Assignment Problem (LAP) is solved at each frame to associate active tracks with current detections. Assuming a one-to-one correspondence between tracks and detections, we seek the assignment that maximizes the overall likelihood of positional measurements, following the approach of IML [57]. This results in the following cost matrix:

$$\mathbf{C}_{ij} = -\log p(\mathbf{z}_t^{\text{pos},j} \mid Z^{\gamma_{1:t-1}^i}), \tag{IV.20}$$

where the likelihoods are provided directly by the Kalman filter. The LAP is then solved using the optimal Jonker-Volgenant algorithm [30]. To avoid erroneous associations between missed tracks and false detections, we introduce a fixed likelihood threshold η and only consider associations for which $p(\mathbf{z}_t^{\text{pos},j} \mid Z^{\gamma_{1:t-1}^i}) > \eta$. Associations below this threshold are discarded.

While this approach is computationally efficient and works well in practice, it remains suboptimal from a Bayesian perspective. A more principled solution—based on Multiple Hypothesis Tracking (MHT)—is developed in Chapter V.

IV.2.5 Optical Flow-Based Velocity Sensor

To provide an independent velocity measurement on frame t, we compute the optical flow between frames I_t and I_{t+1}. It involves finding a function $\boldsymbol{\Phi}_{t,t+1}(\mathbf{w}) \in \mathbb{R}^d$ that represents the apparent motion (displacement) of any pixel $\mathbf{w}$. This is widely used in image registration [140], biomechanical displacement or force measurement [137–139], and key-point tracking through registration [141, 153].

Standard optical flow methods solve an optimization problem that maximizes the similarity between I_t and I_{t+1} after warping, subject to spatial regularization constraints [142, 143, 179]. More recent deep learning approaches treat optical flow as a supervised learning task, typically trained on synthetic datasets [147], or adopt unsupervised learning strategies to directly learn the optimization process [140].

In KOFT, any optical flow algorithm can be employed. Accordingly, we benchmark four different methods in Chapter IV.3.2.

Given an optical flow map, the velocity measurement is extracted at the expected object position:

$$\mathbf{z}_t^{i,\text{vel}} = \mathbf{\Phi}_{t,t+1}\left(\mathbf{H}_{\text{pos}}\hat{\mathbf{x}}_t^{i,\text{SKT}}\right) \tag{IV.21}$$

Unlike detections, velocity measurements are always available and do not require solving an association problem. Indeed, since optical flow relies on pixel intensities and compute pixel-wise displacements, it provides an estimate of the target's velocity, with measurement errors that are largely independent from those of the detection-based measurements. Moreover, it directly measures the forward finite difference $\mathtt{dw}_t$, enabling the Kalman filter to incorporate information from frame $t+1$ when predicting the target's position at time $t+1$ (which traditional Kalman filter are not able to do).

IV.2.6 Track Creation and Termination

After the linking process, detections at time t are either linked to existing tracks or not. New tracks are created from the non-linked detections. To mitigate the impact of false detections, a new track is only created if its initial non-linked detection can be successfully linked over τ_{min} consecutive frames.

To account for situations where an object may be temporarily difficult to detect, non-linked tracks are retained for up to τ_{max} frames: after τ_{max} consecutive missed frames, the track is terminated.

Allowing a track to be re-linked after an extended period of missed detections can lead to erroneous associations as the uncertainty of a non-linked track increases over time in the Kalman filter. We emphasize that in practice, linking a track after $\tau_{\text{max}} > 4$ frames is rare, as the association threshold η effectively prevents the linking of tracks with high uncertainty.

To summarize, our proposed tracking algorithm, KOFT, is an online linking method that integrates linear assignment (LAP) with Kalman filtering. Unlike previous approaches that rely solely on detection-based measurements for linking, KOFT incorporates velocity measurements extracted directly from image data by using optical flow between the current and future frames. As we demonstrate in the following section, this second measurement significantly improves the Kalman filter's robustness to sudden or unpredictable changes in object motion.

IV.3 Results

IV.3.1 Experimental Data

IV.3.1.1 Synthetic Data

We first evaluate our method on synthetic data generated by the SINETRA simulator (see Chapter III). Each synthetic video consists of $T = 200$ frames, each with a resolution of 1024×1024 pixels, and approximately $N \simeq 800$ objects to track. The SINETRA simulator incorporates various motion models in 2D and 3D, and in this chapter, we only evaluate KOFT under the 2D scenarios: Hydra Flow and Springs 2D.

In the Hydra Flow scenario, objects are initially randomly distributed within a segmented mask of a Hydra, which deforms over time to mimic the deformations observed in live animals.

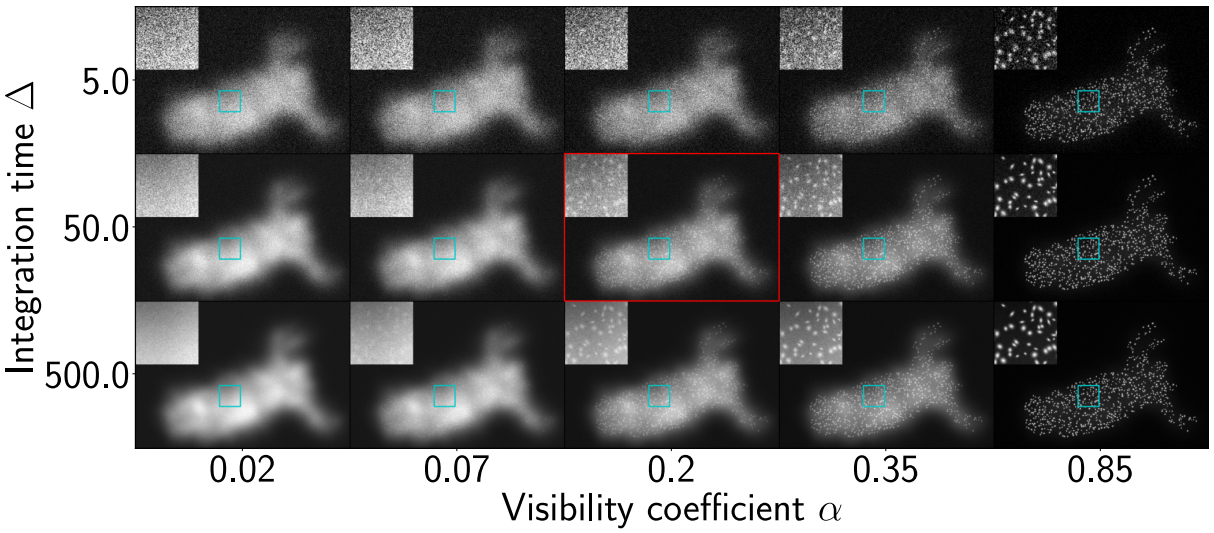


Figure IV.2: Synthetic images Synthetic images generated with the SINETRA simulation framework [65] (Hydra Flow scenario) for different visibility coefficients α and integration times Δ. The default configuration ($\Delta = 50.0, \alpha = 0.2$) is highlighted with a red bounding box

In the Springs 2D scenario, objects are distributed within an ellipsoidal domain that elastically deforms over time, driven by the random motions of spring-connected reference points.

The signal-to-noise ratio (SNR) in these synthetic images is influenced by two parameters: the integration time Δ, which models the Poisson shot noise, and the object visibility coefficient α, which accounts for background intensity (Figure IV.2). Default values are set to $\Delta = 50$ and $\alpha = 0.2$, replicating the SNR observed in experimental fluorescence videos of *Hydra vulgaris* neurons [65]. In Chapter IV.3.4, we investigate the impact of image quality on tracking performance by varying both Δ and α across a broad range of values.

For each experiment, five random videos are generated using the same parameters but different random seeds. In the tables (and figures), we report the mean and standard deviation (for tables) or the 99.7% confidence interval (for figures) over these five random videos.

IV.3.1.2 Hydra Vulgaris Data

Then we validate our method by tracking the neurons of the cnidarian *Hydra vulgaris* within a time-lapse calcium imaging sequence (Figure IV.6-b). The dataset consists of 200 frames with manually partially annotated ground-truth tracks, containing $N = 620$ tracked neurons [10]. Because of manual annotations, tracks may exhibit strong staircasing motions, their position being updated only once every few frames. Therefore, the ground-truth track positions are further smoothed by assuming Gaussian positional noise and applying an optimal Rauch-Tung-Striebel (RTS) smoother [168].

Since neurons are integral part of the animal tissue, spatially close neurons should share similar motions (see left panel of Figure IV.6-b). We found that tracks exhibiting significant deviations from their neighbors' motion were erroneous. To minimize errors introduced by manual annotation of neuron tracks, we assess the deviation between each annotated track and the average motion of its neighbors. Specifically, let $\mathbf{dw}_t^i$ represent the ground truth velocity of track i at frame t, and let $N(i)$ denote the list of the 10 closest neighboring tracks of i. The average velocity of the neighbors is given by:

$$\mathsf{dw}_t^{N(i)} = \frac{1}{10} \sum_{j \in N(i)} \mathsf{dw}_t^j$$

Track i is removed if there exists a frame t such that the deviation in velocity exceeds 1 pixel:

$$\left|\left|\mathsf{dw}_t^i - \mathsf{dw}_t^{N(i)}\right|\right| > 1 \text{ pixels per frame}$$

We manually inspected the tracks removed during preprocessing and found that all but one of the 44 discarded tracks were erroneous. After this cleaning step, a total of $N = 576$ ground-truth tracks remained. We refer to this curated dataset as the *Hydra vulgaris* data in the results section. Despite the cleaning process, this dataset remains flawed for MOT evaluations: many tracks follows mostly undetectable neurons, and several visible neurons are not annotated. In particular, the wavelet detection method [24] which reaches around 80% F1 score on other fully annotated frames [1], only reaches around 30% F1 score here, preventing any consistent linking. Consequently, in the experiments, only the virtual detection process will be used (described in the following section), allowing to measure the linking performances of our algorithms.

In all datasets used in this study, including both synthetic data and real neuronal recordings, the tracked targets exhibit approximately circular shapes with a typical radius of $r \simeq 2$ pixels.

IV.3.2 Optical Flow Selection

Table IV.1: RMSE for different optical flow algorithms on synthetic and experimental datasets. Frame per second (FPS) are computed on the *Hydra vulgaris* dataset, using only the central processing unit of the computer.

Optical flow	**Springs 2D**	**Hydra Flow**	***Hydra vulgaris*** [10]	**FPS**
None	1.41 ± 0.14	1.40 ± 0.23	0.61	/
TV-L^1 [143, 144]	0.79 ± 0.03	$\mathbf{0.50} \pm 0.07$	0.23	8
Farneback [179]	$\mathbf{0.69} \pm 0.03$	0.66 ± 0.02	**0.21**	27
RAFT [147]	1.32 ± 0.11	1.06 ± 0.09	0.33	2
VoxelMorph [140]	1.10 ± 0.03	1.19 ± 0.03	0.40	**40**

KOFT can be implemented with any optical flow algorithm. In this section, we compare several state-of-the-art methods and explain how we select the optical flow algorithm for the tracking experiments that follow.

We evaluate four dense optical flow algorithms on both experimental and synthetic datasets: (1) Farneback [179], that estimates displacement fields by modeling local neighborhoods with polynomial expansions, (2) Total Variation L^1 (TV-L^1) [143, 144], which solves an optimization problem under regularization constraints, (3) Recurrent All-Pairs Field (RAFT) [147], a supervised deep learning approach trained on independent synthetic data, and (4) VoxelMorph [140], an unsupervised deep learning method that we train on our own data, as it does not require annotations.

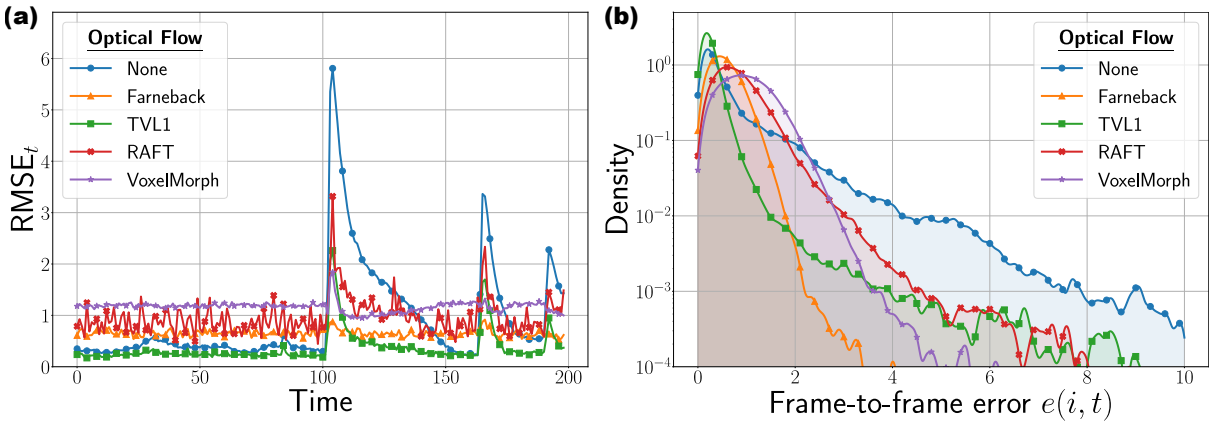


Figure IV.3: Frame-to-frame errors obtained with different optical flow methods on Hydra Flow synthetic videos. In the videos, the animal contracts at times $t = 100, 170, 190$, increasing locally the errors of the different methods. **(a)** Temporal evolution of the Root Mean Squared Errors (RMSE_t) **(b)** Distribution of frame-to-frame errors (log-scaled)

To speed up computations and enhance noise robustness, all images are first smoothed using a Gaussian filter with a $\sigma = 1$ pixel and then downscaled by a factor of 4 (resulting in frames of size 256×256 pixels).

For each algorithm, we tuned the parameters manually as described in the following:

- **TV-L**[1] [143, 144]: We use the OpenCV implementation [181]. We reduced the regularization weight manually to $\lambda = 0.05$ (instead of its default value of 0.15). All other parameters are kept at their default values.

- **Farneback** [179]: We utilize the OpenCV implementation [181]. We adjusted the window size to 20 pixels (instead of its default value of 13) while leaving all other parameters at their defaults.

- **RAFT** [147]: We use the RAFT-small model from the official implementation [147]. Since the pre-trained RAFT model performs poorly with 256×256 images, we downscale the images by a factor of 2 only in that case.

- **VoxelMorph** [140]: We train the unsupervised VoxelMorph model on five different videos: one experimental *Hydra vulgaris* video from [10], two Hydra Flow synthetic videos, and two Springs 2D synthetic videos. The model is trained on randomly sampled frame pairs from these videos until convergence.

The performance of the optical flow algorithms is evaluated by measuring a frame-to-frame optical flow prediction error. For each track i at time t, its ground truth position $\mathbf{w}_t^i$ is propagated to the next frame using the computed optical flow $\mathbf{\Phi}_{t,t+1}(\mathbf{w}_t^i)$ between frames t and $t+1$. The frame-to-frame error is defined as the Euclidean distance between the predicted position $\mathbf{w}_t^i + \mathbf{\Phi}_{t,t+1}(\mathbf{w}_t^i)$ and the ground truth position on the next frame $\mathbf{w}_{t+1}^i$:

$$e(i,t) = \|\mathbf{w}_t^i + \mathbf{\Phi}_{t,t+1}(\mathbf{w}_t^i) - \mathbf{w}_{t+1}^i\|_2 \tag{IV.22}$$

To aggregate errors on each frame, we use a Root Mean Squared Error (RMSE) approach, for each frame $t \leq T-1$, we define the RMSE_t as:

$$\text{RMSE}_t = \sqrt{\frac{1}{N}\sum_{i=1}^{N} e(i,t)^2} \tag{IV.23}$$

Similarly, to aggregate errors for a single video, we use a global RMSE on the video defined as:

$$\text{RMSE} = \sqrt{\frac{1}{N(T-1)}\sum_{i=1}^{N}\sum_{t=1}^{T-1} e(i,t)^2} \tag{IV.24}$$

Table IV.1 summarizes the results for each optical flow algorithm. Across all methods, we observe that the average distance between the optical flow propagated position $\mathbf{w}_t^i + \mathbf{\Phi}_{t,t+1}(\mathbf{w}_t^i)$ and the ground truth position $\mathbf{w}_{t+1}^i$ is smaller than the distance between consecutive ground truth positions $\mathbf{w}_t^i$ and $\mathbf{w}_{t+1}^i$, *i.e.* without optical flow correction (*None* in Table IV.1). For both simulated and experimental data, we also find that TV-L^1 and Farneback outperform the deep learning methods (RAFT and VoxelMorph).

Next, we focus on the errors made by each method in the Hydra Flow simulation, which accounts for the large contraction cycles of *Hydra vulgaris*. These large deformations do not occur in the 200-frame experimental movie that was manually annotated (*Hydra vulgaris* data). We find that Farneback is more resilient to these large contractions compared to the other methods, including TV-L^1 (Figure IV.3-a). Additionally, the distribution of optical flow errors (Figure IV.3-b) shows that large errors (greater than 4 pixels) are less likely with Farneback than with the other methods, which exhibit a longer tail in the error distribution. We also observe that all methods, except TV-L^1, tend to produce small errors (below 1 pixel) when the animal motion is minimal. For instance, outside of the large contractions shown in Figure IV.3-a, VoxelMorph shows a higher RMSE (an increase of 1 pixel on average) compared to when no optical flow is used. This overestimation of animal deformation is likely due to regularization artifacts.

Based on this preliminary comparative study, we select the Farneback algorithm for optical flow implementation in our tracking algorithm. Farneback strikes a good balance between performance and computational efficiency, being 3 to 4 times faster than TV-L^1. Moreover, Farneback is particularly robust to large animal movements, such as Hydra contractions, making it a better choice for tracking under *in vivo* conditions.

IV.3.3 Tracking

IV.3.3.1 Experimental Setup

To measure the accuracy of computed tracks, we use the Higher Order Tracking Accuracy (HOTA) metric [180], which accounts for localization, detection, and association performance (see Chapter III). We use a matching tolerance of r pixels, meaning that a predicted track is considered correctly associated with a ground-truth object only if its position lies within r pixels of the ground truth. In practice, we set $r = 2.0$ pixels, corresponding to the typical size of the targets. Tracks deviating beyond this distance are considered biologically erroneous as they may compromise subsequent analyses.

We compare the performance of KOFT with that of the basic Kalman filter SKT, and KOFT--, where position and velocity are updated simultaneously. Additionally, we evaluate the

performance of two widely-used tracking algorithms: enhanced Multiple Hypothesis Tracking (EMHT) [26] from Icy software [50] and U-TRACK [27] from TrackMate/Fiji software [51, 128].

EMHT is a probabilistic algorithm based on multiple motion models (IMM), which probabilistically handles tracks and uses a multiple hypothesis association method. Due to the high object density in our tracking datasets, we limit the decision tree depth for multiple hypothesis association to $\tau_{\text{depth}} = 3$ frames. U-TRACK is a global distance minimization algorithm solved in two steps (frame-to-frame linking followed by tracklet stitching). We use the advanced version that models motion with Kalman filters [51, 128].

Each method depends on pre-detection of objects. To isolate the impact of the detection method, we implemented an artificial *fake* detector, which uses the ground-truth positions to generate *detections* with known positional noise ($\sigma_{\text{pos}} = 0.5$ pixels) and fixed false negative (FNR) and false positive rates (FPR). The fake detector is denoted as Fake@F1, with FPR = FNR = $1 - \text{F1}$. Additionally, we use the wavelet denoising and thresholding method described in [24], which achieves approximately 80% recall and precision in our synthetic dataset (see the following Section).

All tracking algorithms are parameterized with an association cost limit parameter, η, to prevent the linking of tracks with unlikely detections. For each method, we report the performance metrics obtained with the best threshold η, found through grid search.

The implementations of KOFT and SKT are based on the Python library ByoTrack [22]. Data and code are available at `https://gitlab.pasteur.fr/bia/projects/koft`. All Kalman filters in SKT and KOFT use the following parameters:

- We use the directed motion model, which is suited for this use-case.
- The process noise is set to $\sigma_{\text{mov}} = 1.5$ pixels. This ensures that the Kalman filter expects the velocity of a track to change by no more than $3\sigma_{\text{p}} \simeq 5$ pixels between consecutive frames, which is consistent with motions observed in our experimental datasets.
- The noise on positional and velocity measurements is set to $\sigma_{\text{pos}} = \sigma_{\text{vel}} = r$ pixels, which is the typical size of tracked objects in these experiments.
- We fix $\tau_{\text{min}} = 3$ to be robust to false detections and use $\tau_{\text{max}} = 6$, allowing to bridge over 6 consecutive missed detections.

IV.3.3.2 Tracking Performance on Synthetic Videos

We begin by comparing the performance of tracking algorithms on synthetic videos generated using the SINETRA method [65]. The experiments are conducted using the Springs 2D and Hydra-Flow scenarios (see Chapter IV.3.1.1). Fluorescent spot detection is performed with a wavelet denoising and thresholding algorithm [24], resulting in F1-scores of $88.9 \pm 1.4\%$ for the Springs 2D scenario and $84.1 \pm 2.0\%$ for the Hydra-Flow scenario. The lower detection accuracy in the Hydra-Flow scenario is attributed to the frequent contractions of the animal, which cause spot aggregation.

In Table IV.2, we present the HOTA metric for the different tracking algorithms. We observe that incorporating optical flow estimation of object velocity significantly improves tracking performance in both KOFT-- and KOFT. The latter outperforms the former thanks to its two-step track state update. In the Springs 2D scenario (and similarly in the Hydra-Flow scenario),

Table IV.2: Tracking performance (HOTA at 2 pixels tolerance [180]) on synthetic videos. Frame per second (FPS) are computed on the same laptop and do not account for the detection process.

Scenario	Springs 2D	Hydra Flow	FPS
U-TRACK (Fiji) [27,51,128]	$80.4 \pm 4.3\%$	$69.0 \pm 3.5\%$	25
EMHT (Icy) [26,50]	$82.9 \pm 2.8\%$	$75.8 \pm 5.3\%$	2
SKT (ours)	$80.5 \pm 3.6\%$	$72.8 \pm 3.8\%$	25
KOFT-- (ours)	$90.2 \pm 1.8\%$	$84.6 \pm 2.5\%$	10
KOFT (ours)	$\mathbf{92.5} \pm 1.5\%$	$\mathbf{87.9} \pm 1.9\%$	10

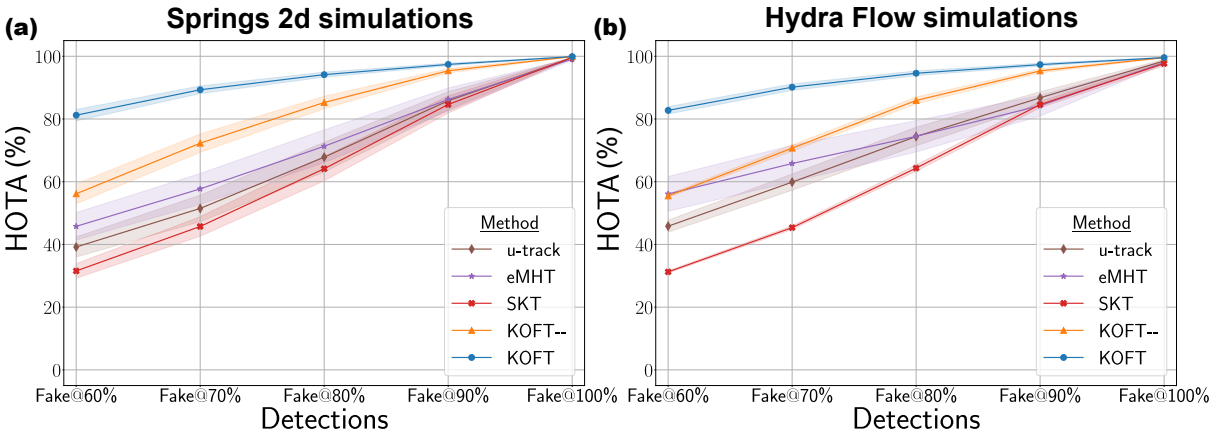


Figure IV.4: Tracking performance on synthetic videos. The HOTA metric [180] with 2 pixels tolerance is reported with respect to detection accuracy (Fake@F1). The light-colored envelope around each curve corresponds to the 99.7% confidence interval over M = 5 simulations. **(a)** Results on Springs 2D synthetic videos **(b)** Results on Hydra Flow synthetic videos

KOFT achieves 92.5% HOTA (respectively 87.9% HOTA), whereas the best tracking algorithm without optical flow velocity updates, EMHT, achieves only 82.9% HOTA (respectively 75.8% HOTA). Additionally, we observe that rapid animal contractions result in more tracking errors, with a consistent drop in tracking performance (approximately a 5% decrease in HOTA) in the Hydra-Flow scenario compared to the Springs 2D scenario.

Regarding computational load, U-TRACK and SKT are significantly faster than the other methods, achieving 25 frames per second (FPS) in our configuration. Despite the inclusion of optical flow computation (using the FARNEBACK [179] method from OpenCV [181], which runs at 27 FPS), both KOFT and KOFT-- maintain relatively efficient performance at 10 FPS. In comparison, EMHT, which employs a multiple hypothesis framework for linking detections to tracks, runs at a lower frame rate (2 FPS).

To isolate the impact of the detection method and assess the robustness of the tracking algorithms, we use the artificial *fake* detector (see Chapter IV.3.3.1) and evaluate the performance of the tracking algorithms under varying levels of detection error (Figure IV.4). Among standard tracking algorithms, i.e., those without optical flow estimation of object velocity, we first note that U-TRACK [27,51,128] is slightly more accurate than SKT. Since both algorithms are based on the same Bayesian filter, this slight difference likely arises from U-TRACK's better handling of track creation, association, and termination compared to SKT. However, both algorithms are

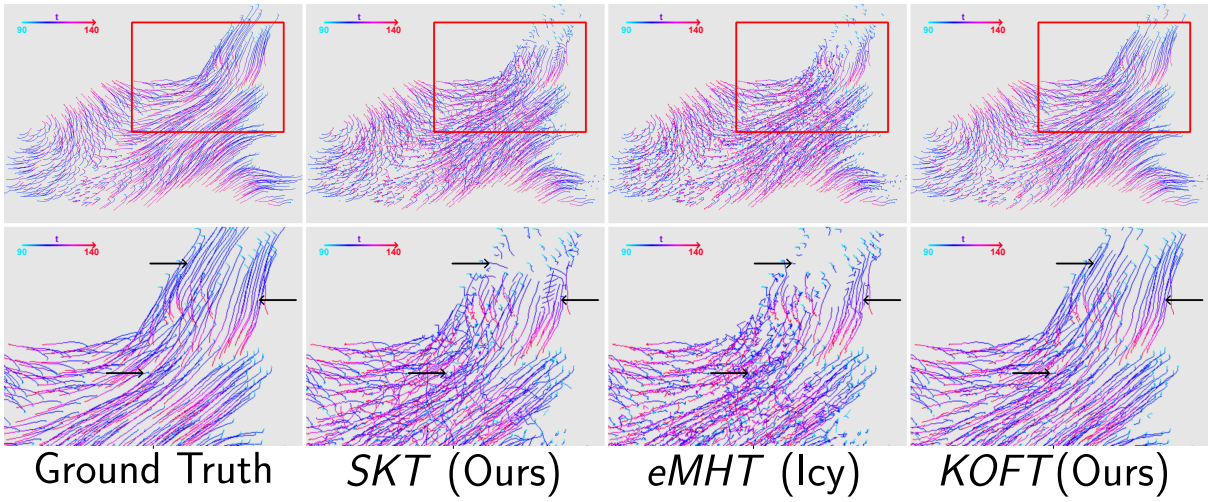


Figure IV.5: Temporal projection of tracks from a Hydra-Flow synthetic video spanning 50 frames, centered around a contraction event (frames 90 to 140, with the contraction occurring at $t = 100$). Detection is performed using wavelet denoising and thresholding [24]. Black arrows highlight a selection of tracks during the animal's contraction, where KOFT successfully maintains track consistency, while standard methods (SKT and EMHT) fail.

outperformed by EMHT [26, 50], especially in conditions with poor detection quality (Fake@ less than 80%), owing to the multiple hypothesis association implemented in EMHT.

In these experiments, we again observe that updating object velocity using optical flow estimation in KOFT and KOFT-- leads to improved tracking accuracy, in particular with low-quality detection. Notably, KOFT maintains HOTA above 80% across all detection accuracies, effectively reducing tracking errors by nearly a factor of two in all conditions. For example, errors with standard tracking algorithms, i.e., those without optical flow velocity estimation, exceed 25% at Fake@80%, whereas errors with KOFT remain below 10% in both the Springs 2D and Hydra-Flow simulations.

In the KOFT-- algorithm, velocity is updated only for linked tracks through a single Kalman update that incorporates both position and velocity. In contrast, KOFT updates both linked and non-linked tracks with optical flow velocity estimations, thanks to its dual-step update that decouples position and velocity updates (see Chapter IV.2.2). This approach makes KOFT more robust to missed object detections, significantly improving tracking performance under low detection accuracy (Fake@ below 80%). For instance, in the Hydra-Flow scenario at Fake@70%, KOFT-- achieves 70.7% HOTA, while KOFT maintains a higher HOTA of 90.1%.

To visualize the errors made by the different tracking algorithms, we temporally project object tracks in a Hydra-Flow simulation (see Figure IV.5). As the animal contracts, tracking becomes particularly challenging in the animal's tentacles, located in the top-right corner of the projection. In this region, object acceleration reaches up to 10 pixels per frame^2 at time $t = 100$. All standard algorithms, which do not update velocity using optical flow, either lose tracks or incorrectly associate them. In contrast, the tracks produced by KOFT remain spatially consistent and are much more resilient to these sudden motions.

This initial set of tracking experiments using synthetic simulations demonstrates that incorporating object velocity estimation and updates via optical flow significantly enhances tracking accuracy compared to standard Bayesian filtering, across various simulation scenarios and a

wide range of detection accuracies.

IV.3.3.3 Tracking Neurons in Hydra Vulgaris

To compare KOFT with other tracking algorithms on experimental data, we use fluorescence calcium imaging of the cnidarian *Hydra vulgaris* and manually annotated neuron tracks over 200 frames (20 s at 10 Hz imaging; see section IV.3.1.2).

In Figure IV.6-a, we report tracking accuracy (HOTA at 2 pixels tolerance) for various tracking methods as a function of increasing detection quality (Fake@F1; see section IV.3.3.1).

The standard filtering algorithms, U-TRACK and SKT, are the least performant on this experimental dataset, with HOTA starting at $\sim$ 38% for SKT and $\sim$ 50% for U-TRACK at Fake@60%, and remaining below $\sim$ 80% up to Fake@80%. The multiple hypothesis EMHT performs better, with HOTA increasing from $\sim$ 60% (Fake@60%) to $\sim$ 90% (Fake@80%). Notably, these performances slightly outperform those obtained in synthetic simulations for the Hydra Flow scenario (Figure IV.4-b). For instance, HOTA scores for U-TRACK and EMHT increase by 5.2% and 17.1%, respectively, with Fake@80% detections in the experimental dataset compared to the simulations. This improvement likely results from the smaller frame-to-frame deformations of the animal in the experimental data compared to simulations: the frame-to-frame RMSE without optical flow correction (referred to as *None* in Table IV.1) is 1.40 pixels for Hydra Flow sequences versus 0.61 pixels for the experimental *Hydra vulgaris* data.

The incorporation of optical flow estimation of object velocity further improves tracking performance. The KOFT HOTA increases from $\sim$ 83% (Fake@60%) to $\sim$ 100% (Fake@100%). Notably, at Fake@80%, KOFT makes half as many errors (4% error) as EMHT (8% error). Figure IV.6-b illustrates how our method successfully tracks neurons in *Hydra vulgaris*, including in challenging regions such as the densely packed foot and the highly dynamic tentacles. This underscores the effectiveness of KOFT for experimental data, particularly for tracking neurons in live animals.

IV.3.4 Impact of imaging noise on optical flow and KOFT accuracy

The performance of KOFT relies heavily on the accuracy of optical flow estimates for animal deformation and object velocity. In this section, we investigate how image quality affects optical flow estimations and KOFT accuracy by varying the parameters α (object visibility coefficient) and Δ (integration time) in the synthetic movie generator SINETRA [65] (Figure IV.2).

First, we assess the impact of image quality on optical flow estimation using the FARNEBACK algorithm (Figure IV.7-a). The frame-to-frame RMSE between the ground-truth deformation and optical flow estimation increases with higher Poisson shot noise (lower Δ) and lower visibility coefficients (α), *i.e.*, when the target brightness is low relative to the background intensity. For low values of α and Δ (e.g., $\alpha \leq 0.3$ for $\Delta = 5.0$ and $\alpha \leq 0.1$ for $\Delta = 50.0$), we observe that the RMSE after optical flow correction exceeds that of the no-correction baseline ($\Phi = 0$). This suggests that, for very noisy images, using optical flow to update object velocity may actually impair tracking accuracy. As we will show below, the σ_{vel} parameter in our KOFT algorithm accounts for this risk and mitigates its impact.

We next evaluate the impact of image quality and the resulting optical flow degradation on the tracking accuracy (HOTA metric) of KOFT (Figure IV.7-b). To disentangle the effects of image quality on optical flow estimation and object detection, we use an artificial detector and set the detection quality to Fake@80%. The Kalman filter framework of KOFT accounts for

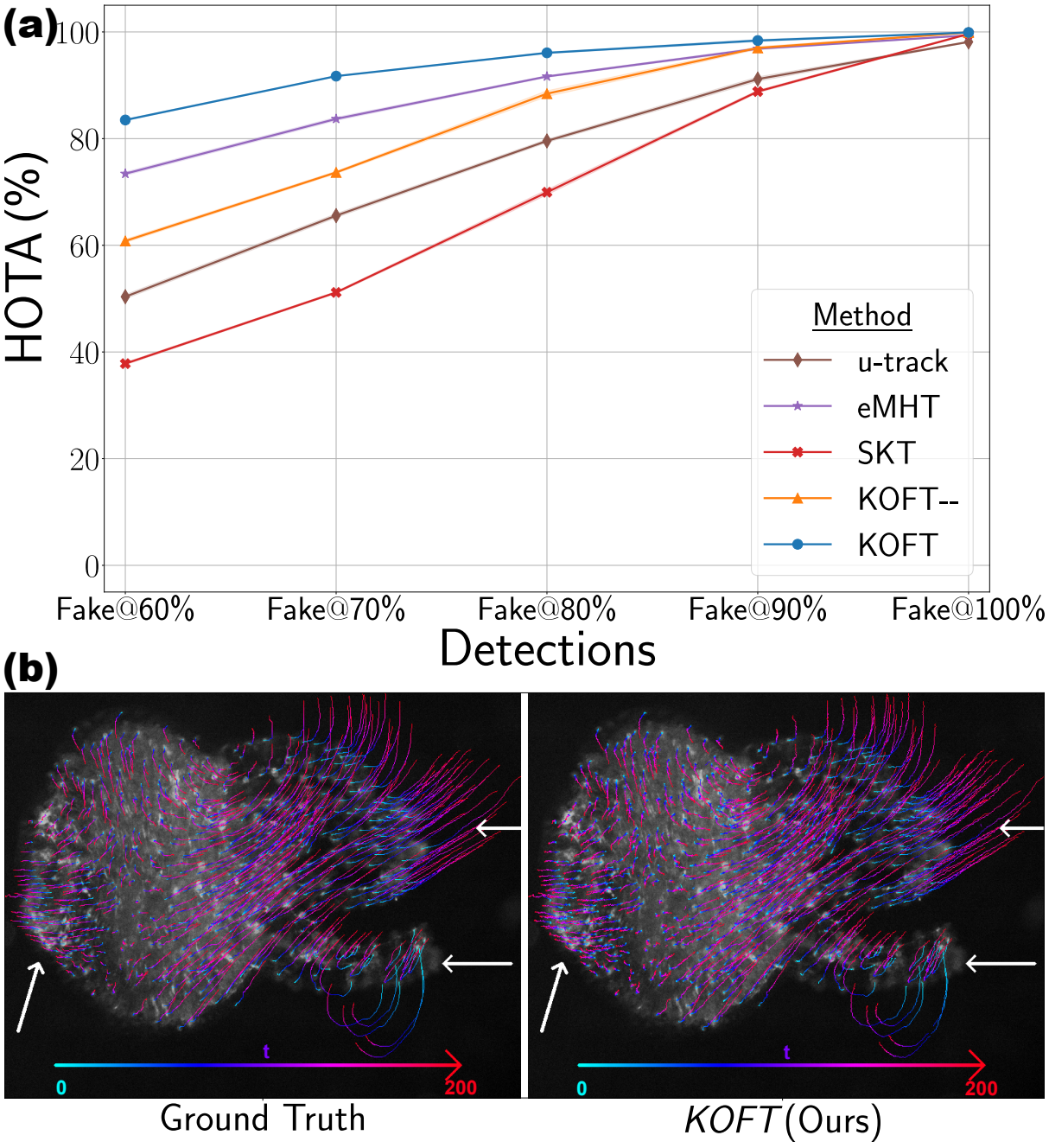


Figure IV.6: Tracking performance on the *Hydra vulgaris* experimental dataset. (a) HOTA metric comparison for different tracking algorithms as a function of detection quality (Fake@F1). **(b)** Temporal projections of tracks overlaid on the first frame of the video. The Hydra begins elongating at frame 30 and continues until frame 200. The cleaned ground-truth (left) and the KOFT (right) tracks are shown for comparison. KOFT successfully tracks most neurons, including those in challenging regions of the animal such as the foot and tentacles (indicated by white arrows).

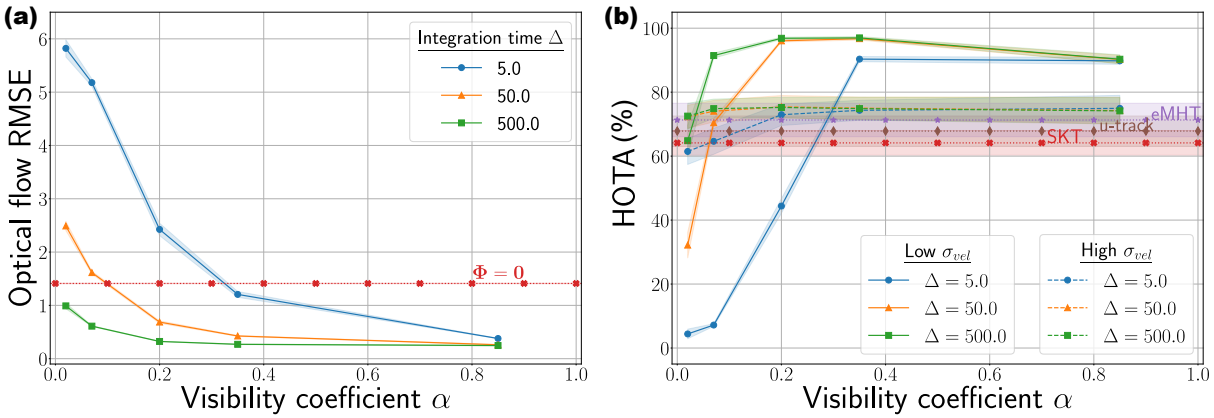


Figure IV.7: Optical flow (Farneback) and KOFT accuracy for different image qualities. (a) Frame-to-frame RMSE of optical flow computed using the FARNEBACK algorithm [179] on synthetic images (Springs 2D scenario) for increasing object visibility coefficient α. Three different integration times, $\Delta = 5.0$ (blue), $\Delta = 50.0$ (orange), and $\Delta = 500.0$ (green), are considered. The RMSE without optical flow correction ($\Phi = 0$) is displayed as a red baseline. **(b)** Tracking performance (HOTA at 2-pixel tolerance [180]) of KOFT on Springs 2D synthetic videos, for increasing object visibility coefficient α and three different values of integration time Δ. We use a *fake* detector (Fake@80%). Two implementations of KOFT are compared: one with high confidence in the measured velocities (lower $\sigma_{\text{vel}} = 1$, solid lines) and one with low confidence (higher $\sigma_{\text{vel}} = 5$, dashed lines). The light-colored envelope around each curve represents the 99.7% confidence interval over $M = 5$ simulations.

the uncertainty in optical flow estimation under noisy conditions by incorporating the standard deviation σ_{vel} of velocity estimates (Eq. IV.15). By default, σ_{vel} is set to 2 pixels per frame but can be adjusted based on image noise and the accuracy of optical flow estimates.
We consider two scenarios:
1. A low uncertainty scenario with $\sigma_{\text{vel}} = 1.0$ pixel per frame, suited for high image quality and accurate optical flow estimation.
2. A high uncertainty scenario with $\sigma_{\text{vel}} = 5.0$ pixels per frame, adapted to noisy conditions where optical flow computation may be erroneous.
In both cases, we observe that tracking performance (*HOTA*) decreases with increased imaging noise (lower α and Δ values) and the associated degradation of optical flow estimates.

When using a large uncertainty ($\sigma_{\text{vel}} = 5.0$ pixels per frame), KOFT shows increased resilience to imaging noise and consistently outperforms SKT, except under the most extreme imaging conditions ($\alpha = 0.02$ and $\Delta = 5.0$). Notably, even when the frame-to-frame RMSE is worse than the baseline without optical flow correction ($\Phi = 0$) (as shown in red in Figure IV.7-a), KOFT successfully incorporates noisy flow estimates into the track states, outperforming the baselines. For instance, it achieves 72.9% HOTA at $\alpha = 0.2$ and $\Delta = 5.0$, outperforming SKT, U-TRACK, and EMHT, despite an optical flow RMSE of 2.43 pixels (approximately 70% larger than without optical flow correction).

In contrast, for high image quality and accurate optical flow estimates, KOFT with large uncertainty ($\sigma_{\text{vel}} = 5.0$ pixels per frame) shows limited performance. For example, with the default configuration ($\alpha = 0.2$ and $\Delta = 50.0$) and a low RMSE for optical flow (0.69 ± 0.03), KOFT with high uncertainty achieves 75.4% HOTA, whereas it reaches 96.0% HOTA with a lower $\sigma_{\text{vel}} = 1.0$, reflecting higher confidence in velocity updates. However, in this high-confidence scenario ($\sigma_{\text{vel}} = 1$), noisy flow estimates significantly decrease tracking performance

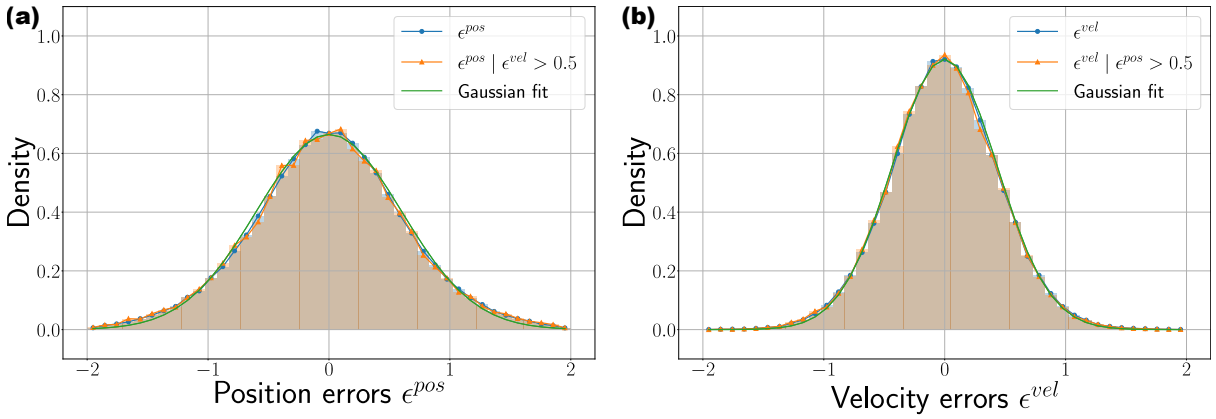


Figure IV.8: Error distributions for position and velocity sensors on a synthetic Springs 2D video. A Gaussian fit (green) is overlaid for comparison. Conditional error distributions are also shown to assess the independence assumption. **(a)** Position error distribution. **(b)** Velocity error distribution.

when $\alpha \leq 0.1$ and $\Delta < 50$.

These experiments demonstrate that KOFT outperforms standard tracking solutions across a wide range of noise levels, particularly when σ_{vel} is adjusted to match the image quality and the accuracy of optical flow estimates. For practical purposes, we find that the default setting of $\sigma_{\text{vel}} = 2.0$ provides a good balance between performance and robustness.

IV.3.5 Sensor Independence

In KOFT, we assume that the position and velocity measurements, denoted by $\mathbf{z}_t^{\text{pos}}$ and $\mathbf{z}_t^{\text{vel}}$ respectively, are statistically independent for each track at time t. To empirically validate this assumption, we use the SINETRA dataset [65] and apply the Wavelet detection algorithm [24] to obtain position measurements and the Farneback optical flow algorithm [179] to estimate velocities.

For each ground-truth track i, we have access to its true position $\mathbf{w}_t^i$ and true velocity $\mathsf{d}\mathbf{w}_t^i$ (computed via forward finite differences) at every frame $t < T$. The measurement errors for position and velocity are defined as:

$$\boldsymbol{\epsilon}_t^{i,\text{pos}} = \mathbf{w}_t^i - \mathbf{z}_t^{i,\text{pos}}, \quad \boldsymbol{\epsilon}_t^{i,\text{vel}} = \mathsf{d}\mathbf{w}_t^i - \mathbf{z}_t^{i,\text{vel}}. \tag{IV.25}$$

In our Kalman filtering framework (Chapter IV.2.2), these error terms are modeled as independent Gaussian variables. To test this assumption, we compute the empirical cross-correlation matrix $\mathbf{C} \in \mathbb{R}^{d \times d}$:

$$\mathbf{C} = \frac{1}{N(T-1)} \sum_{i=1}^{N} \sum_{t=1}^{T-1} \boldsymbol{\epsilon}_t^{i,\text{pos}} \cdot \boldsymbol{\epsilon}_t^{i,\text{vel}\mathsf{T}}. \tag{IV.26}$$

If the independence assumption holds, the entries of $\mathbf{C}$ should be close to zero.

We compute $\mathbf{C}$ on both the Springs 2D and Hydra flow scenarios and find that the maximum empirical correlation across all entries is $\max(\mathbf{C}) = 0.003$, observed between the x-components of position and velocity. This is notably smaller than the correlations observed within each modality (e.g., 0.005 between x and y components of position, and 0.01 for velocity), and negligible compared to the measurement noise variances σ_{pos}^2 and σ_{vel}^2.

Figure IV.8 shows the distributions of the position and velocity errors in the Springs 2D scenario using the Wavelet and FARNEBACK algorithms. As shown, these distributions are approximately Gaussian, and the conditional distributions $\boldsymbol{\epsilon}^{\text{vel}} \mid \boldsymbol{\epsilon}^{\text{pos}} > 0.5$ and $\boldsymbol{\epsilon}^{\text{pos}} \mid \boldsymbol{\epsilon}^{\text{vel}} > 0.5$ closely matches the marginal distribution, further supporting the independence assumption.

IV.4 Discussion

In this chapter, we introduce KOFT, a novel multi-sensor tracking approach that enhances Multiple Object Tracking (MOT) in noisy fluorescence images, particularly under challenging conditions of high object density and abrupt motion changes. A key innovation of KOFT is the use of dense optical flow as an additional sensor for estimating object velocity, which complements standard position measurements from object detections. This velocity information is integrated into a Kalman filtering framework, enabling the system to anticipate sudden or irregular object displacements that traditional models, relying solely on past positions, struggle to predict. By combining both detection and optical flow measurements, KOFT improves tracking accuracy and robustness. The resulting displacement prediction represents an optimal trade-off between motion continuity inferred from past frames and real-time velocity observations derived from optical flow.

We benchmarked KOFT against several state-of-the-art tracking algorithms using synthetic data from the SINETRA simulation framework [65]. Our results demonstrate that KOFT significantly enhances tracking performance across a wide range of scenarios, reducing tracking errors by at least a factor of two. We also validated the performance of KOFT on real-world calcium imaging data of *Hydra vulgaris* neuronal activity. This application is particularly challenging due to the animal's large and sudden contractions. The ability to track neurons in behaving animals is a critical step in understanding neuronal activity in response to environmental cues [28, 182]. Given these challenges, KOFT represents a substantial improvement in the robust monitoring of neuronal activity across various animal models.

The accuracy of optical flow for estimating object motion is influenced by imaging conditions, such as fluorescence noise. As expected, we observed that the tracking performance of KOFT deteriorates when optical flow fails to reliably predict object motion. To mitigate the risk of erroneous motion estimation and tracking errors, the velocity measurement noise, σ_{vel}, is a key parameter in our Kalman filter model (Eq IV.15). By increasing σ_{vel}, the confidence in the optical flow measurements is reduced, which enhances the robustness of KOFT under noisy conditions.

In KOFT, detection associations are iteratively solved frame by frame through an efficient likelihood maximization process. More complex and principled techniques, such as multiple hypothesis tracking (MHT), with a complete Bayesian formulation [26] should further improve accuracy (see Chapter V). However, they would come at a higher computational cost. Alternatively, integrating interactive multiple models (IMM) [57] could increase robustness by enabling automatic selection of the most efficient Kalman filter model, further enhancing performance under challenging conditions.

Chapter V

KOFT-MHT: Bayesian and Multiple Hypothesis Tracking Framework

In the previous chapter, we demonstrated the benefits of incorporating optical flow as an auxiliary sensor in Kalman filter-based trackers. We introduced the SKT and KOFT algorithms, two frame-by-frame online linking methods (see Chapter II.1.3.3) that rely on Kalman filtering (augmented with optical flow in KOFT) to compute association likelihoods, followed by a maximum-likelihood solution using the Hungarian algorithm.

In this chapter, we extend this idea to a fully principled Bayesian tracking framework based on Multiple Hypothesis Tracking (MHT), which explicitly accounts for *a-priori* in data association. We evaluate our approach on the synthetic datasets introduced in Chapter III, and show that our Bayesian implementation substantially improves tracking accuracy, particularly under noisy detection conditions.

This work will be submitted at the IEEE ISBI 2026 conference.

V.1 Introduction

Bayesian trackers, and in particular Multiple Hypothesis Trackers (MHT), have demonstrated strong robustness in biological object tracking, often outperforming frame-by-frame and distance-based approaches [26, 132, 134].

In the previous chapter, we introduced a novel Bayesian motion model based on Kalman filtering enhanced with optical flow. We initially integrated this model into a frame-by-frame linking algorithm, KOFT, that maximizes the association likelihoods (and *de facto* ignore association priors). KOFT already surpassed more complex Bayesian trackers on Hydra-like datasets, owing to the additional motion information provided by the optical-flow-enhanced Kalman filter.

We hypothesize, however, that incorporating this enhanced motion model into a fully principled Bayesian tracker would yield further improvements. To test this, we developed a Bayesian tracking framework grounded in Track-Oriented MHT [129, 131] and proposed two new algorithms. The first integrates standard Kalman filtering into the Bayesian framework, serving as a validation of our MHT implementation; we refer to this tracker as the SKT-MHT. The second builds on the same framework but replaces the motion model with our optical-flow-enhanced Kalman filter; we call this method KOFT-MHT.

In this chapter, we first describe the design of our Bayesian tracking framework, with particular emphasis on user-friendly a-priori parameters that can be set with minimal expertise. We then leverage the SINETRA synthetic simulator to evaluate our framework and compare SKT-MHT and KOFT-MHT against state of the art biological MHT approaches and our previous frame-by-frame algorithms, SKT and KOFT. Finally, we investigate the influence of key parameters, including the MHT tree depth and the few a priori probabilities required by our Bayesian formulation.

Our experiments show that the Bayesian framework equipped with standard Kalman filtering already outperforms other MHT implementations using comparable motion models. Furthermore, when combined with the optical-flow-enhanced model, our Bayesian tracker significantly outperforms all alternatives on SINETRA, including KOFT. Importantly, we also demonstrate that our implementation is robust to parameter choices, enabling practical use even with approximate manual tuning by non-expert users.

V.2 Bayesian Multiple Object Tracking

Symbol	Description
t, τ	Temporal indices
i, j	Target index, Detection index
T	Number of frames in the video
d	Spatial dimension of images (Usually, $d \in \{2, 3\}$ for 2D and 3D images)
$I_{1:T}$	$= (I_1, \dots, I_T)$, a video of T images (frames)
N	Number of targets in the video
M_t	Number of detections on frame t
$d_{\mathbf{x}}$	Dimension of the state space used for targets
$d_{\mathbf{z}}$	Dimension of the measurement space (a common case is $d_{\mathbf{z}} = d$, when measurements are positions)
$\mathbf{x}_t^i$	$\in \mathbb{R}^{d_{\mathbf{x}}}$, state vector of the i^{th} target at time t.
$\mathbf{z}_t^j$	$\in \mathbb{R}^{d_{\mathbf{z}}}$, measurement vector of the j^{th} detection at time t
X_t	$= (\mathbf{x}_t^1, \dots, \mathbf{x}_t^N)$, states of all the targets at time t
$X_{1:T}$	$= (X_1, \dots, X_T)$, states of all the targets on all the frames
Z_t	$= (\mathbf{z}_t^1, \dots, \mathbf{z}_t^{M_t})$, all the detections measurements at time t
$Z_{1:T}$	$= (Z_1, \dots, Z_T)$, all the detections
J_t	$= [\![1,\, M_t]\!]$, detections indices at time t
$\mathsf{P}^1(J_t)$	$= \{\gamma_t \subset J_t \mid \lvert\gamma_t\rvert \leq 1\}$, set of all the singletons of J_t and the empty set $\emptyset$.
$\Omega(T)$	$\bigotimes_{t=1}^{T} \mathsf{P}^1(J_t)$, set of all feasible associations between targets and detections under the 1-to-1 association assumption
γ_t^i	$\subset J_t$, set of the associated detections indices to the i^{th} target at time t
$\gamma_{1:T}^i$	$\in \Omega(T)$, all the associations for the i^{th} target ("track"). Abbreviated γ^i.
$\Gamma_{1:T}$	$= (\gamma^1, \dots, \gamma^N) \subset \Omega(T)$, the set of tracks (track-oriented notations).

$\Omega(t+\tau \mid \Gamma_{1:t})$	$\subset \Omega(t+\tau)$, feasible tracks up to frame $t+\tau$ that matches with confirmed tracks $\Gamma_{1:t}$ up to frame t.
$H_t(\tau)$	$\subset \Omega(t+\tau \mid \Gamma_{1:t})$, considered association hypotheses in the MHT framework.
$t^i_{\mathrm{b}}, t^i_{\mathrm{d}}$	$\in [\![0, T]\!]$, the first (birth) and last (death) frame of appearance of the target i.
$J^{\mathrm{b}}_t \cup J^{\mathrm{f}}_t \cup J^{\mathrm{a}}_t$	$= J_t$, partition of the M_t detections on frame t into M^{b}_t that initiate new tracks, M^{f}_t false positives and M^{a}_t associated detections to a previously existing tracks.
Γ^{c}_t	$\subset [\![1, N]\!]$, active "candidate" track indices on frame t (*i.e.* $t^i_{\mathrm{b}} \leq t \leq t^i_{\mathrm{d}}$)
$\Gamma^{\mathrm{d}} \cup \Gamma^{\mathrm{m}} \cup \Gamma^{\mathrm{a}} \cup \Gamma^{\mathrm{b}}$	$= \Gamma^{\mathrm{c}}_t$, partition of the N^{c}_t candidate tracks into N^{d}_t tracks that terminates, N^{m}_t miss-detected tracks, N^{a}_t associated tracks and N^{b}_t newly initiated tracks.
p_{d}	$\in [0, 1]$, target termination probability
p_{b}	$\in [0, 1]$, target birth rate
p_{m}	$\in [0, 1]$, miss-detection probability
p_{f}	$\in [0, 1]$, false positive track probability
λ_{b}	$\in \mathbb{N}$, expected number of birth per frame
λ_{f}	$\in \mathbb{N}$, expected number of false positives per frame
η	$\in \mathbb{R}^+$, association threshold below which assignments are not considered
$\mathbb{P}(\cdot), p(\cdot)$	Discrete probability and continue density distribution
$\mathcal{B}(\,\cdot\,; p)$	Bernoulli distribution with probability of success p
$\mathcal{B}(\,\cdot\,; p, n)$	Binomial distribution with n trials and probability of success p
$\mathcal{N}(\,\cdot\,; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$
$\mathcal{P}(\,\cdot\,; \lambda)$	Poisson distribution distribution with mean λ

In this Section, we first recall the Bayesian tracking framework introduced in Chapter II. We then introduce the notations and derive the factorization of the posterior tracking objective, before presenting our resulting Multiple Hypothesis Tracking (MHT) solution.

In Bayesian Multiple Object Tracking, each target i is modeled by a hidden state vector $\mathbf{x}^i_t \in \mathbb{R}^{d_{\mathbf{x}}}$ that evolves over time. The total number of targets N is unknown and must be inferred jointly with their states. The goal of a Bayesian tracker is to estimate the most probable joint sequence of states

$$X_{1:T} = (\mathbf{x}^i_t)_{\substack{1 \leq i \leq N \\ 1 \leq t \leq T}}$$

given the observed video frames $I_{1:T}$. This can be formulated as a Maximum A Posteriori (MAP) estimation:

$$X^\star_{1:T} \in \arg\max_{X_{1:T}} \mathbb{P}(X_{1:T} \mid I_{1:T}) \tag{V.1}$$

In practice, modeling the full image generation process is extremely challenging due to background clutter, imaging noise, occlusions, texture variations, and depth information, which may not be modeled in the state vectors. Some works such as [58] attempt to address this by

modeling targets as Gaussian spots with additive background, but such methods remain rare and may struggle with more complex images.

Instead, most practical Bayesian tracking algorithms adopt a *tracking-by-detection* paradigm, wherein a detector is first applied to the images to extract a set of measurements:

$$Z_{1:T} = \text{Detector}(I_{1:T}) \tag{V.2}$$

Each Z_t is an arbitrarily ordered set of detection vectors $\mathbf{z}_t^j \in \mathbb{R}^{d_\mathbf{z}}$ encoding positional and appearance features (e.g. location, bounding box, shape, intensity, deep visual descriptors). The Bayesian tracking problem then reduces to:

$$X_{1:T}^\star \in \arg\max_{X_{1:T}} \mathbb{P}(X_{1:T} \mid Z_{1:T}) \tag{V.3}$$

where detections serve as proxies for the true observations. Despite this simplification, evaluating the likelihood $\mathbb{P}(Z_{1:T} \mid X_{1:T})$ remains difficult, as it requires to model both the measurement process (the detector) and resolving ambiguous data associations — *i.e.*, determining which detection corresponds to which target.

To mitigate this, most BMOT approaches introduce a latent *association variable* $\Gamma_{1:T}$ encoding the assignment of detections to tracks (formally detailed below). The MAP problem is then decoupled into two subproblems:

1. **Association estimation:**

$$\Gamma_{1:T}^\star \in \arg\max_{\Gamma_{1:T}} \mathbb{P}(\Gamma_{1:T} \mid Z_{1:T}) \tag{V.4}$$

2. **State estimation:**

$$X_{1:T}^\star \in \arg\max_{X_{1:T}} \mathbb{P}(X_{1:T} \mid \Gamma_{1:T}^\star, Z_{1:T}) \tag{V.5}$$

In the following sections, we first introduce the necessary notations (Section V.2.1) that allow us to reformulate Equation V.4 into a factorized form (Section V.2.2). This reformulation provides the foundation for our proposed Multiple Hypothesis Tracking (MHT) solver (Section V.2.3). We then present simple estimators that reduce the number of free parameters and simplify the overall parametrization of the method (Section V.2.4). The resulting MHT solver is independent of the choice of motion model. We therefore reuse the motion models developed in Chapter IV, and specify the key details required to integrate them into our MHT framework (Section V.2.5). Finally, we address the state estimation problem (Eq. V.5), a step that is often overlooked, and provide a principled solution (Section V.2.6).

V.2.1 Notation

We use i to index tracks, j for detections, and t for frames. Each frame t contains M_t detections indexed by $J_t = [\![1,\, M_t]\!]$. The full sequence of detection index sets is $J_{1:T} = (J_t)_{1 \le t \le T}$.

A track for target i is represented as $\gamma_{1:T}^i = (\gamma_t^i)_{1 \le t \le T}$, where:

$$\forall t \in [\![1,\, T]\!]\,, \gamma_t^i \subset J_t \tag{V.6}$$

and γ_t^i is the subset of detections assigned to target i at time t. This formalism is illustrated in Figure V.1.

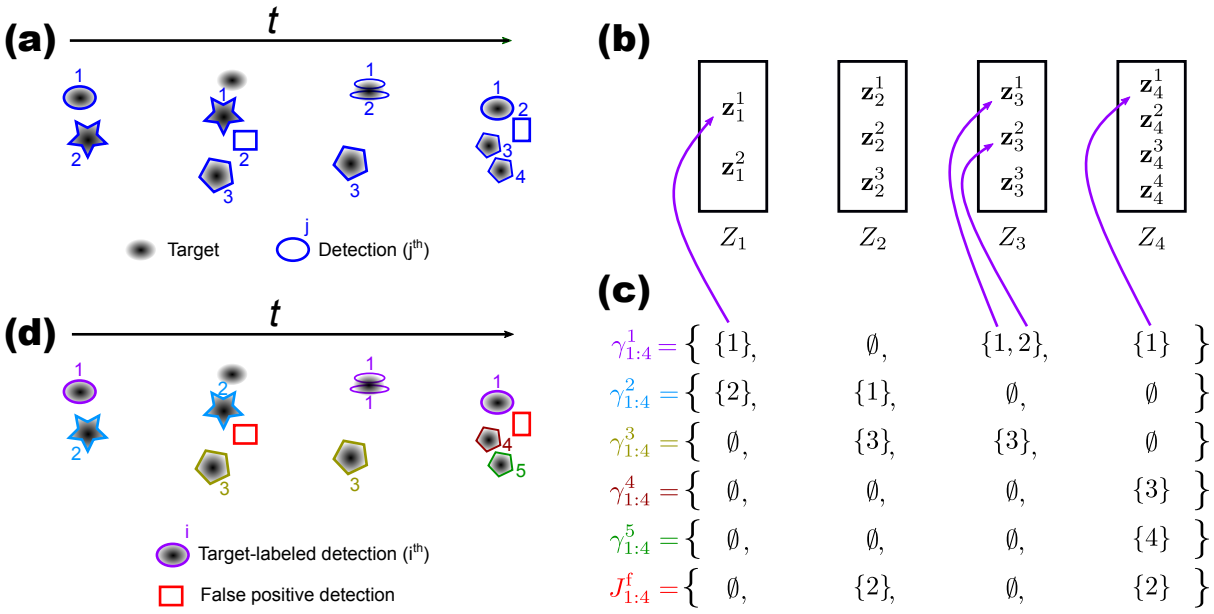


Figure V.1: Illustration of the Bayesian tracking framework formalism (repeated for clarity from Chapter II, with additional precisions), highlighting key challenges in object tracking: false positives, missed detections, and target/detection splitting.
(a) Detections on four frames are shown, with arbitrary indexing from top to bottom. The circle target is missed on frame 2, and false positive detections (shown as rectangles) appear on frames 2 and 4. On frame 3, two detections (1 and 2) correspond to the same circle target (i.e., a split detection).
(b) Detections sets $Z_{1:4}$ corresponding to the top-left panel.
(c) True labeling of detections using our track-oriented formalism. Each track is associated with unique color (see on bottom left panel), while red denotes the set of false positives. For instance, the track 1 (circle, purple) starts on frame 1 with the detection first detection $\mathbf{z}_1^1$ ($\gamma_1^1 = \{1\}$), it is miss-detected on frame 2 ($\gamma_2^1 = \emptyset$), then associated to the two first detections of frame 3 ($\gamma_3^1 = \{1, 2\}$) and finally associated to $\mathbf{z}_4^1$ ($\gamma_4^1 = \{1\}$). Note that if our notations allow split detection, our proposed MHT solver assumes the 1-to-1 association between tracks and detections, and will have to associate track 1 with a single detection on frame 3, while classifying the other as false positive. The matching between γ^1 and detections is illustrated by purple arrows. A birth event occurs for the pentagon target on frame $t_{\mathrm{b}}^3 = 2$, and a death event for the star target on frame $t_{\mathrm{d}}^2 = 2$. On frame 4, the pentagon undergoes a target splitting event, producing two new targets (4 and 5).
(d) Detections are shown with their associated track identifier. Again, each color represents a unique target, while red denotes false positives: *e.g.*, detections 2 on frames 2 and 4 are unassigned. On frame 2, the detection indices $J_2 = \{1, 2, 3\}$ are partitioned into $J_2^{\mathrm{f}} = \{2\}, J_2^{\mathrm{b}} = \{3\}$ and $J_2^{\mathrm{a}} = \{1\}$, indicating that the first detection (star) is associated, the second (rectangle) is false, and the third (pentagon) correspond to a track initiation. Similarly, the candidate tracks $\Gamma_2^{\mathrm{c}} = \{1, 2, 3\}$ are partitioned into $\Gamma_2^{\mathrm{d}} = \emptyset, \Gamma_2^{\mathrm{m}} = \{1\}, \Gamma_2^{\mathrm{a}} = \{2\}$ and $\Gamma_2^{\mathrm{b}} = \{3\}$, indicating that the first track (circle) is missed, the second track (star) is associated, and the third track (pentagon) was born on this frame.

This general formalism allows modeling of missed detections, false positives, track births and deaths, as well as split and merged detections. However, for factorization and optimization purposes (see Chapter V.2.2), we impose the common *1-to-1 association* constraint: each detection is assigned to at most one track, and each track to at most one detection per frame. This common constraint removes the possibility of modeling merged or split detections. When a detection corresponds to multiple true targets (e.g., overlapping cells in a single bounding box), only one target may be associated with it—the others are treated as missed. Conversely, when a single target generates multiple detections (e.g., an over-segmented cell), only one detection is retained, and the others are classified as false positives.

The set of all feasible tracks with the detections indices $J_{1:T}$ is denoted $\Omega(T)$, which can be written as a cartesian product:

$$\Omega(T) = \bigotimes_{t=1}^{T} \mathsf{P}^1(J_t) \tag{V.7}$$

where $\mathsf{P}^1(J_t) = \{\emptyset, \{1\}, \ldots, \{M_t\}\}$ is the set containing all the singletons of J_t and the empty set $\emptyset$.

For each track $\gamma^i_{1:T}$, its *birth* and *death* times are defined as its first and last appearance times (see tracks 2 and 3 from Figure II.2):

$$t^i_{\mathrm{b}} = \min\{t \mid |\gamma^i_t| \geq 1\} \tag{V.8}$$

$$t^i_{\mathrm{d}} = \max\{t \mid |\gamma^i_t| \geq 1\} \tag{V.9}$$

The set of detection vectors associated with track i at time t is denoted:

$$Z_t^{\gamma^i_t} = \{\mathbf{z}^j_t \mid j \in \gamma^i_t\} \tag{V.10}$$

and the full sequence across time is $Z^{\gamma^i_{1:T}} = (Z_t^{\gamma^i_t})_{1 \leq t \leq T}$.

A linking algorithm aims to construct N tracks over T frames, denoted $\Gamma_{1:T} = (\gamma^i_{1:T})_{1 \leq i \leq N} \subset \Omega(T)$, that best explain the detections $Z_{1:T}$ according to Eq. V.4. Given such tracks $\Gamma_{1:T}$, the set of detection indices J_t on each frame t can be partitioned into three disjoint subsets:

$$J_t = J^{\mathrm{b}}_t \ \cup \ J^{\mathrm{f}}_t \ \cup \ J^{\mathrm{a}}_t,$$

where:

- J^{b}_t contains the M^{b}_t detections that initiate new tracks on frame t:

$$J^{\mathrm{b}}_t = \left\{ j \in J_t \;\middle|\; \exists\, i,\ \gamma^i_t = \{j\} \text{ and } t^i_{\mathrm{b}} = t \right\}, \tag{V.11}$$

- J^{f}_t contains the M^{f}_t false-positive detections on frame t:

$$J^{\mathrm{f}}_t = \left\{ j \in J_t \;\middle|\; \forall i,\ j \notin \gamma^i_t \right\}, \tag{V.12}$$

- J^{a}_t contains the M^{a}_t detections that are associated with tracks already existing before frame t:

$$J^{\mathrm{a}}_t = \left\{ j \in J_t \;\middle|\; \exists\, i,\ \gamma^i_t = \{j\} \text{ and } t^i_{\mathrm{b}} < t \right\}. \tag{V.13}$$

By construction, the number of detections at frame t satisfies

$$M_t = M_t^{\mathrm{b}} + M_t^{\mathrm{f}} + M_t^{\mathrm{a}}.$$

While the death time of a track is well-defined over the full T frames, it may be ambiguous when only considering the truncated sequence $\Gamma_{1:t}$. For example, $\gamma_{1:3}^i = \{\{1\}, \{2\}, \emptyset\}$ could mean either that track i terminates at frame 2, or that it is temporarily missed and will reappear later. To resolve this ambiguity, we introduce a latent variable $\Gamma_t^{\mathrm{d}} \subset [\![1,\, N]\!]$, the set of tracks that have definitively terminated between frames $t-1$ and t (i.e., $t_{\mathrm{d}}^i = t-1$ for $i \in \Gamma_t^{\mathrm{d}}$).

We also define $\Gamma_t^{\mathrm{c}} \subset [\![1,\, N]\!]$ as the set of *candidate* tracks at frame t, i.e., tracks that were born before t and are still potentially active (not yet marked as dead). Let $N_t^{\mathrm{c}} = |\Gamma_t^{\mathrm{c}}|$ denote their count. By definition,

$$\Gamma_t^{\mathrm{d}} \subset \Gamma_t^{\mathrm{c}} \subset [\![1,\, N]\!] \setminus \bigcup_{\tau<t} \Gamma_\tau^{\mathrm{d}},$$

since a track can die at most once and dead tracks do not compete for associations.

Among candidate tracks, we define:

- $\Gamma_t^{\mathrm{m}} \subset \Gamma_t^{\mathrm{c}} \setminus \Gamma_t^{\mathrm{d}}$: tracks that survive but are missed at frame t, with $N_t^{\mathrm{m}} = |\Gamma_t^{\mathrm{m}}|$. These account for false negative detections.
- Γ_t^{a}: the remaining surviving tracks that are associated with a detection, with $N_t^{\mathrm{a}} = N_t^{\mathrm{c}} - N_t^{\mathrm{d}} - N_t^{\mathrm{m}}$.

Under the 1-to-1 association assumption, we have $M_t^{\mathrm{a}} = N_t^{\mathrm{a}}$, and there exists a mapping $\mathcal{L}_t = \{(i, \gamma_t^i),\ i \in \Gamma_t^{\mathrm{a}}\}$ between Γ_t^{a} and J_t^{a}.

For compactness, we also introduce mutually exclusive binary indicator variables:

- $\delta_t^{i,\bar{\mathrm{c}}} = 1 \Leftrightarrow i \notin \Gamma_t^{\mathrm{c}}$ (non-candidate tracks),
- $\delta_t^{i,\mathrm{d}} = 1 \Leftrightarrow i \in \Gamma_t^{\mathrm{d}}$ (track i dies between $t-1$ and t),
- $\delta_t^{i,\mathrm{m}} = 1 \Leftrightarrow i \in \Gamma_t^{\mathrm{m}}$ (missed detected track at t),
- $\delta_t^{i,\mathrm{a}} = 1 \Leftrightarrow i \in \Gamma_t^{\mathrm{a}}$ (associated track at t),
- $\delta_t^{i,\mathrm{b}} = 1 \Leftrightarrow \gamma_t^i \subset J_t^{\mathrm{b}}$ (track birth at t).

V.2.2 Posterior Factorization

A key step in solving Eq. V.4 is to factorize the posterior distribution $\mathbb{P}(\Gamma_{1:T} \mid Z_{1:T})$. We extend here the factorization of Reid [130] (which did not model track termination) to the notation and models introduced in this work. Observe that $(\Gamma_{1:T}, Z_{1:T})$ fully determines both $\Gamma_{1:T}^{\mathrm{d}}$ and $J_{1:T}^{\mathrm{f}}$. Thus, Eq. V.4 can be rewritten as:

$$\begin{aligned}
\Gamma_{1:T}^{\star} &= \underset{\Gamma_{1:T} \subset \Omega(T)}{\arg\max}\ \mathbb{P}(\Gamma_{1:T} \mid Z_{1:T}) \\
&= \underset{\Gamma_{1:T} \subset \Omega(T)}{\arg\max}\ \mathbb{P}(\Gamma_{1:T}, \Gamma_{1:T}^{\mathrm{d}}, J_{1:T}^{\mathrm{f}} \mid Z_{1:T}) \\
&= \underset{\Gamma_{1:T} \subset \Omega(T)}{\arg\max}\ \mathbb{P}(\Gamma_{1:T}, \Gamma_{1:T}^{\mathrm{d}}, J_{1:T}^{\mathrm{f}})\, \mathbb{P}(Z_{1:T} \mid \Gamma_{1:T}, J_{1:T}^{\mathrm{f}}),
\end{aligned} \tag{V.14}$$

where $\mathbb{P}(\Gamma_{1:T}, \Gamma^{\mathrm{d}}_{1:T}, J^{\mathrm{f}}_{1:T})$ is the *prior* on association events, and $\mathbb{P}(Z_{1:T} \mid \Gamma_{1:T}, J^{\mathrm{f}}_{1:T})$ is the *likelihood* of the observed detections given those associations. The latter does not depend on $\Gamma^{\mathrm{d}}_{1:T}$ since track terminations do not affect the measurement generation model.

Likelihood factorization. Under the standard assumption that targets are conditionally independent given the associations, neglecting tracks interactions, the likelihood decomposes as:

$$\begin{aligned}\mathbb{P}(Z_{1:T} \mid \Gamma_{1:T}, J^{\mathrm{f}}_{1:T}) &= \mathbb{1}_{\Gamma^{1:T}, J^{\mathrm{f}}_{1:T}}(Z_{1:T}) \prod_{i=1}^{N} p(Z^{\gamma^i_{1:T}}) \times \prod_{t=1}^{T} \prod_{j \in J^{\mathrm{f}}_t} p^{\mathrm{f}}(\mathbf{z}^j_t) \\ &= \mathbb{1}_{\Gamma^{1:T}, J^{\mathrm{f}}_{1:T}}(Z_{1:T}) \prod_{i=1}^{N} \prod_{t=1}^{T} p(Z^{\gamma^i_t}_t \mid Z^{\gamma^i_{1:t-1}}) \times \prod_{t=1}^{T} \prod_{j \in J^{\mathrm{f}}_t} p^{\mathrm{f}}(\mathbf{z}^j_t)\end{aligned} \tag{V.15}$$

where $\mathbb{1}_{\Gamma_{1:T}, J^{\mathrm{f}}_{1:T}}$ enforces cardinality consistency between the detections and the association variables. The single-target likelihood density $p(Z^{\gamma^i_t}_t \mid Z^{\gamma^i_{1:t-1}})$ is typically given by a state-space filtering model (e.g., Kalman or particle filtering) and defaults to 1 when $\gamma^i_t = \emptyset$.

Prior factorization. Applying the chain rule over time, the prior can be written as:

$$\mathbb{P}(\Gamma_{1:T}, \Gamma^{\mathrm{d}}_{1:T}, J^{\mathrm{f}}_{1:T}) = \prod_{t=1}^{T} \mathbb{P}(\Gamma_{1:t}, \Gamma^{\mathrm{d}}_t, J^{\mathrm{f}}_t \mid \Gamma_{1:t-1}, \Gamma^{\mathrm{d}}_{1:t-1}, J^{\mathrm{f}}_{1:t-1}), \tag{V.16}$$

where each factor represents the prior on tracks prolongation from frame $t-1$ to t.

At frame t, given past associations, the prolongation event can be decomposed into the following latent choices:

1. Select the subset $\Gamma^{\mathrm{d}}_t \subset \Gamma^{\mathrm{c}}_t$ of *candidate* tracks that terminate (die) from $t-1$ to t. We assume an independent and constant death probability p_{d} for each candidate track:

$$\mathbb{P}(\Gamma^{\mathrm{d}}_t \mid \Gamma_{1:t-1}, \Gamma^{\mathrm{d}}_{1:t-1}, J^{\mathrm{f}}_{1:t-1}) = p_{\mathrm{d}}^{N^{\mathrm{d}}_t} (1 - p_{\mathrm{d}})^{N^{\mathrm{c}}_t - N^{\mathrm{d}}_t},$$

2. Select the subset Γ^{m}_t of surviving candidate tracks that are missed at t. Surviving tracks can be missed with independent and constant probability p_{m}:

$$\mathbb{P}_{|t-1}(\Gamma^{\mathrm{m}}_t \mid \Gamma^{\mathrm{d}}_{1:t}, \Gamma_{1:t-1}, J^{\mathrm{f}}_{1:t-1}) = p_{\mathrm{m}}^{N^{\mathrm{m}}_t} (1 - p_{\mathrm{m}})^{N^{\mathrm{a}}_t},$$

3. The remaining candidate tracks Γ^{a}_t are associated to detections; this fixes $M^{\mathrm{a}}_t = N^{\mathrm{a}}_t = |\Gamma^{\mathrm{a}}_t|$.

4. Draw the number of new track births M^{b}_t and false positives M^{f}_t. We assume the commonly used independent Poisson prior with parameters λ_{b} and λ_{f}, respectively:

$$\mathbb{P}(M^{\mathrm{b}}_t) = \frac{\lambda_{\mathrm{b}}^{M^{\mathrm{b}}_t}}{M^{\mathrm{b}}_t!} e^{-\lambda_{\mathrm{b}}}, \quad \mathbb{P}(M^{\mathrm{f}}_t) = \frac{\lambda_{\mathrm{f}}^{M^{\mathrm{f}}_t}}{M^{\mathrm{f}}_t!} e^{-\lambda_{\mathrm{f}}}.$$

5. Given $M_t = M^{\mathrm{a}}_t + M^{\mathrm{b}}_t + M^{\mathrm{f}}_t$, partition the detection indices J_t into:
   - the set J^{b}_t of birth detections among M_t,
   - the set J^{f}_t of false detections among the remaining,

- and the remaining are the associated detections J_t^{a}.

Finally, choose the mapping $\mathcal{L}_t$ from J_t^{a} to Γ_t^{a}. As the set of detections are arbitrarily ordered, these yield a Uniform distribution on the number of existing configuration:

$$\mathbb{P}(J_t^{\mathrm{b}}, J_t^{\mathrm{f}}, J_t^{\mathrm{a}}, \mathcal{L}_t | M_t, \Gamma_t^{\mathrm{a}}) = \frac{1}{\binom{M_t}{M_t^{\mathrm{b}}}\binom{M_t - M_t^{\mathrm{b}}}{M_t^{\mathrm{f}}} M_t^{\mathrm{a}}!}$$

Combining these terms, the prolongation prior probability simplifies to:

$$\mathbb{P}(\Gamma_{1:t}, \Gamma_t^{\mathrm{d}}, J_t^{\mathrm{f}} \mid \Gamma_{1:t-1}, \Gamma_{1:t-1}^{\mathrm{d}}, J_{1:t-1}^{\mathrm{f}}) = p_{\mathrm{d}}^{N_t^{\mathrm{d}}}(1-p_{\mathrm{d}})^{N_t^{\mathrm{c}} - N_t^{\mathrm{d}}} p_{\mathrm{m}}^{N_t^{\mathrm{m}}}(1-p_{\mathrm{m}})^{N_t^{\mathrm{a}}} \lambda_{\mathrm{b}}^{M_t^{\mathrm{b}}} \lambda_{\mathrm{f}}^{M_t^{\mathrm{f}}} \frac{e^{-\lambda_{\mathrm{b}} - \lambda_{\mathrm{f}}}}{M_t!} \quad \text{(V.17)}$$

Therefore, the prior factorizes into:

$$\mathbb{P}(\Gamma_{1:T}, \Gamma_{1:T}^{\mathrm{d}}, J_{1:T}^{\mathrm{f}}) = \prod_{t=1}^{T} \frac{e^{-\lambda_{\mathrm{b}} - \lambda_{\mathrm{f}}}}{M_t!} \prod_{i=1}^{N} g_t^i \prod_{j \in J_t^{\mathrm{f}}} \lambda_{\mathrm{f}}, \quad \text{(V.18)}$$

$$\text{with } g_t^i = p_{\mathrm{d}}^{\delta_t^{i,\mathrm{d}}} \left[(1-p_{\mathrm{d}})p_{\mathrm{m}}\right]^{\delta_t^{i,\mathrm{m}}} \left[(1-p_{\mathrm{d}})(1-p_{\mathrm{m}})\right]^{\delta_t^{i,\mathrm{a}}} \lambda_{\mathrm{b}}^{\delta_t^{i,\mathrm{b}}}. \quad \text{(V.19)}$$

Final posterior form. Substituting the likelihood and prior into Eq. V.14 yields:

$$\Gamma_{1:T}^{\star} \in \underset{\Gamma_{1:T} \subset \Omega(T)}{\arg\max} \prod_{t=1}^{T} \prod_{i=1}^{N} G_t^i \prod_{j \in J_t^{\mathrm{f}}} \lambda_{\mathrm{f}}\, p^{\mathrm{f}}(\mathbf{z}_t^j), \quad \text{(V.20)}$$

$$\text{with } G_t^i = p_{\mathrm{d}}^{\delta_t^{i,\mathrm{d}}} \left[(1-p_{\mathrm{d}})p_{\mathrm{m}}\right]^{\delta_t^{i,\mathrm{m}}} \left[(1-p_{\mathrm{d}})(1-p_{\mathrm{m}})\, p(Z_t^{\gamma_t^i} \mid Z^{\gamma_{1:t-1}^i})\right]^{\delta_t^{i,\mathrm{a}}} \left[\lambda_{\mathrm{b}}\, p(Z_t^{\gamma_t^i})\right]^{\delta_t^{i,\mathrm{b}}}. \quad \text{(V.21)}$$

V.2.3 MHT Solution

The set of feasible tracks $\Omega(T)$ grows exponentially with time, making exact optimization intractable. Thus, identifying the optimal subset of tracks $\Gamma_{1:T}$ requires approximate strategies. Multiple Hypothesis Tracking (MHT) [26,129–134] addresses this by solving associations recursively over short temporal windows of size $\tau_{\mathrm{depth}} \ll T$. Let $\Gamma_{1:t}$ denote the set of confirmed tracks up to frame t. For a look-ahead horizon τ, the set of feasible track continuations is

$$\Omega(t+\tau \mid \Gamma_{1:t}) = \left\{\gamma_{1:t+\tau} \in \Omega(t+\tau) \,:\, \gamma_{1:t} \in \Gamma_{1:t} \cup \{(\emptyset, \ldots, \emptyset)\}\right\},$$

where the all-empty prefix corresponds to tracks born between $t{+}1$ and $t{+}\tau$. At each iteration MHT solves

$$\Gamma_{1:t+\tau_{\mathrm{depth}}}^{\star} \in \underset{\Gamma_{1:t+\tau_{\mathrm{depth}}} \subset \Omega(t+\tau_{\mathrm{depth}} | \Gamma_{1:t}^{\star})}{\arg\max} \mathbb{P}(\Gamma_{1:t+\tau_{\mathrm{depth}}} \mid Z_{1:t+\tau_{\mathrm{depth}}}), \quad \text{(V.22)}$$

then advances the time index by $\tau_{\mathrm{step}} \leq \tau_{\mathrm{depth}}$ frames and repeats.

We adopt the *Track-Oriented MHT* (TOMHT) framework [26, 131, 132], where each active track at frame t is recursively *branched* into multiple feasible continuations, referred to as *hypotheses*. Over successive frames, hypotheses grow into trees of depth $1+\tau_{\mathrm{depth}}$ (Figure V.2), effectively exploring a subset $H_t(\tau_{\mathrm{depth}}) \subset \Omega(t + \tau_{\mathrm{depth}} \mid \Gamma_{1:t})$ that contains only plausible prolongations.

Importantly, the choice of hypotheses must also resolve ambiguities: whether a candidate track is still active, has terminated, or whether its detections correspond instead to false positives. To this end, each hypothesis $\gamma_{1:t+\tau}^i \in H_t(\tau)$ is associated with three probability factors:

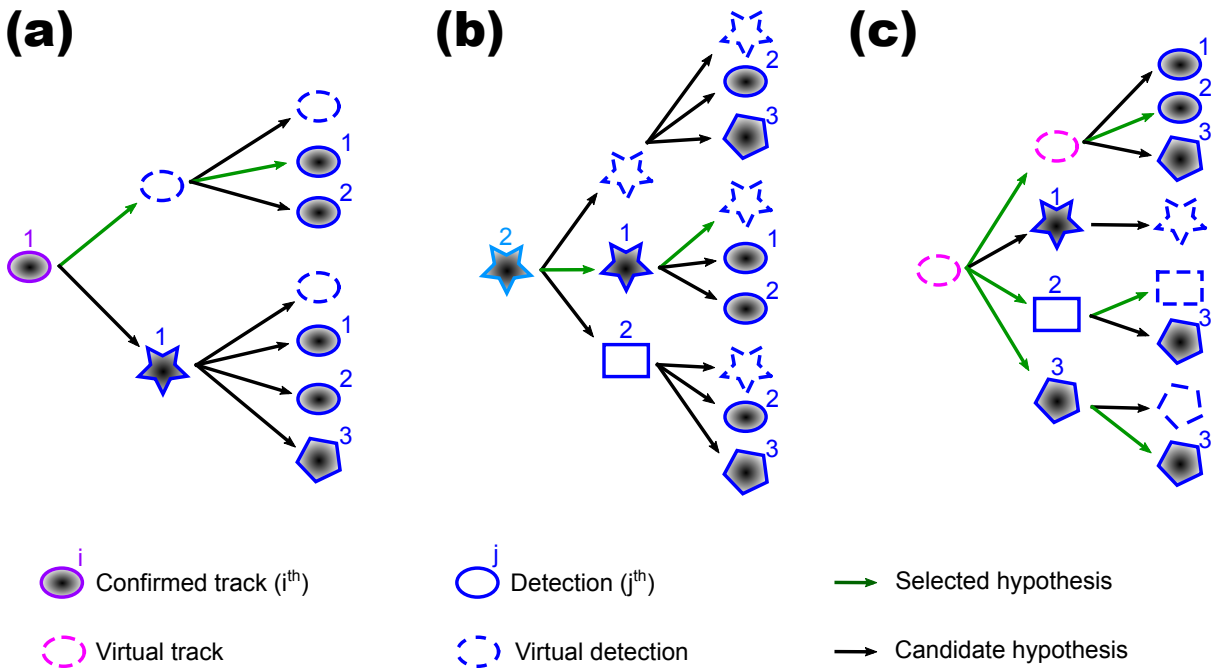


Figure V.2: Illustration of our hypotheses trees in MHT, applied to the tracking scenario from Figure V.1 (repeated from Chapter IV for clarity, with additional precisions). Hypothesis trees are initiated at frame $t = 1$ from two confirmed tracks (circle and star), and generated with a depth $\tau_{\text{depth}} = 2$. The 1-to-1 optimal association corresponds to the selected hypotheses in green. Note that virtual tracks and detections corresponds to $\emptyset$ our track-oriented formalism.
(a) Hypothesis tree for the circle track. With a tree-depth of 2, the association decision is made once the target has re-appeared on frame 3. The optimal solution in green correspond to the hypothesis $\{\{1\}, \emptyset, \{1\}\} \in H_1(2)$.
(b) Hypothesis tree for the star track. The track is associated to virtual detection as it terminates on frame 3. The optimal solution in green correspond to the hypothesis $\{\{2\}, \{1\}, \emptyset\} \in H_1(2)$.
(c) Track initiation hypotheses, from a virtual track. The three selected hypotheses in green are: $\{\emptyset, \emptyset, \{2\}\}$, $\{\emptyset, \{2\}, \emptyset\}$ and $\{\emptyset, \{3\}, \{3\}\}$. Two tracks are initiated in frame 2 (rectangle and pentagon). The rectangle track will be classified as false positive. The 1-to-1 constraint also impose to create a track for the split detection of the circle target on frame 3, which will be classified as false positive later on.

1. **Active:** the hypothesis is a continuing track at frame $t + \tau$,

$$G^{\text{c}}(\gamma^i_{1:t+\tau}) = \prod_{u=t^i_{\text{b}}}^{t+\tau} G^i_u.$$

2. **Terminated:** the hypothesis ended at $t^i_{\text{d}} \leq t + \tau$,

$$G^{\text{d}}(\gamma^i_{1:t+\tau}) = \prod_{u=t^i_{\text{b}}}^{t^i_{\text{d}}} G^i_u.$$

3. **False positive:** all detections in the hypotheses are instead classified as false,

$$G^{\text{f}}(\gamma^i_{1:t+\tau}) = \prod_{\mathbf{z}^j_u \in Z^{\gamma^i_{1:t+\tau}}} \lambda_{\text{f}}\, p^{\text{f}}(\mathbf{z}^j_u).$$

Selecting a hypothesis corresponds to choosing its most probable factor, thereby resolving ambiguities. Any valid subset $\Gamma_{1:t+\tau}$ must cover all detections exactly once, with false positives explained by false tracks that are penalized through their probability factors. Using the posterior factorization (Eq. V.20), the optimization (V.22) becomes:

$$\Gamma^{\star}_{1:t+\tau} \in \underset{\Gamma_{1:t+\tau}\subset H_t(\tau)}{\arg\max}\ \mathbb{1}_{\Gamma_{1:t+\tau}}(Z_{1:t+\tau}) \prod_{\gamma^i_{1:t+\tau}\in\Gamma_{1:t+\tau}} \max\{G^{\mathrm{c}}, G^{\mathrm{f}}, G^{\mathrm{d}}\}(\gamma^i_{1:t+\tau}), \tag{V.23}$$

where $\mathbb{1}_{\Gamma_{1:t+\tau}}(Z_{1:t+\tau})$ enforces the one-to-one association between detections and tracks, with every detection associated.

V.2.3.1 Hypotheses Generation

Given current tracks $\Gamma_{1:t}$, hypotheses $H_t(\tau) \subset \Omega(t+\tau \mid \Gamma_{1:t})$ are generated recursively to depth τ_{depth}:

$$H_t(0) = \Gamma_{1:t}, \qquad H_t(\tau+1) = \mathcal{H}\big(H_t(\tau)\big),\ \ \tau = 0, \ldots, \tau_{\mathrm{depth}} - 1, \tag{V.24}$$

where $\mathcal{H}$ applies 3 branching rules:

1. **Association:** For every $\gamma^i_{1:t+\tau} \in H_t(\tau)$ and detection $\mathbf{z}^j_{t+\tau+1}$, a prolongation hypothesis is generated:

   $$(\gamma^i_{1:t+\tau}, \{j\}) \in H_t(\tau+1),$$

   and the following recursive probability updates are applied:

   $$G^{\mathrm{c}}(\gamma^i_{1:t+\tau+1}) = G^{\mathrm{c}}(\gamma^i_{1:t+\tau})(1-p_{\mathrm{d}})(1-p_{\mathrm{m}})p(\mathbf{z}^j_{t+\tau+1}|Z^{\gamma^i_{1:t+\tau}}),$$

   $$G^{\mathrm{d}}(\gamma^i_{1:t+\tau+1}) = G^{\mathrm{c}}(\gamma^i_{1:t+\tau+1})p_{\mathrm{d}}, \qquad G^{\mathrm{f}}(\gamma^i_{1:t+\tau+1}) = G^{\mathrm{f}}(\gamma^i_{1:t+\tau})\lambda_f\, p^{\mathrm{f}}(\mathbf{z}^j_t)$$

2. **Miss:** For each $\gamma_{1:t+\tau} \in H_t(\tau)$, its empty prolongation is added:

   $$(\gamma_{1:t+\tau}, \emptyset) \in H_t(\tau+1),$$

   with recursive probability updates

   $$G^{\mathrm{c}}(\gamma^i_{1:t+\tau+1}) = G^{\mathrm{c}}(\gamma^i_{1:t+\tau})(1-p_{\mathrm{d}})p_{\mathrm{m}},$$

   $$G^{\mathrm{d}}(\gamma^i_{1:t+\tau+1}) = G^{\mathrm{d}}(\gamma^i_{1:t+\tau}), \qquad G^{\mathrm{f}}(\gamma^i_{1:t+\tau+1}) = G^{\mathrm{f}}(\gamma^i_{1:t+\tau}).$$

3. **Birth:** For each detection $\mathbf{z}^j_{t+\tau+1}$, generate a new track hypothesis (Figure II.6.c)

   $$(\emptyset, \ldots, \emptyset, \{j\}) \in H_t(\tau+1),$$

   initialized with

   $$G^{\mathrm{c}}(\gamma^i_{1:t+\tau+1}) = \lambda_{\mathrm{b}}\, p(\mathbf{z}^j_t),$$

   $$G^{\mathrm{d}}(\gamma^i_{1:t+\tau+1}) = G^{\mathrm{c}}(\gamma^i_{1:t+\tau+1})p_{\mathrm{d}}, \qquad G^{\mathrm{f}}(\gamma^i_{1:t+\tau+1}) = \lambda_f\, p^{\mathrm{f}}(\mathbf{z}^j_t)$$

V.2.3.2 Cost Minimization

Let $n_t = |H_t(\tau_{\text{depth}})|$ be the number of hypotheses considered and $m_t = |\Gamma^\star_{1:t}| + \sum_{\tau=1}^{\tau_{\text{depth}}} M_{t+\tau}$, the number of confirmed tracks and following detections in the look-ahead horizon τ_{depth}. A binary vector $\mathbf{y} \in \{0,1\}^{n_t}$ defines a subset of selected tracks $\Gamma_{1:t+\tau_{\text{depth}}} \subset H_t(\tau_{\text{depth}})$. Let $\mathbf{c} \in \mathbb{R}^{n_t}$ be the cost vector associated with each hypothesis such that $\mathbf{c}^i = -\log\max\{G^{\text{c}}, G^{\text{f}}, G^{\text{d}}\}(\gamma^i_{1:t+\tau_{\text{depth}}})$. By definition of the probability factors, the selected tracks have a log-posterior $\mathbf{y}^\mathsf{T}\mathbf{c}$. Finally, a binary matrix $\mathbf{A} \in \{0,1\}^{n_t \times m_t}$ is defined such that A_{kl} indicates that the k^{th} hypothesis is using the l^{th} track or detection. The final optimization problem can then be written as:

$$\begin{aligned} &\text{minimize} && \mathbf{y}^\mathsf{T}\mathbf{c} \\ &\text{subject to} && \mathbf{A}^\mathsf{T}\mathbf{y} = \mathbf{1}, \quad \mathbf{y} \in \{0,1\}^{n_t} \end{aligned} \tag{V.25}$$

where $\mathbf{A}^\mathsf{T}\mathbf{y} = \mathbf{1}$ enforces the 1-to-1 association constraint.

Problem (V.25) is a multidimensional LAP and hence a Mixed Integer Linear Program (MILP). In practice, we use the HiGHs solver [109] through SciPy [183], which we found faster than the Gurobi Python wrapper [111].

V.2.3.3 Hypotheses Pruning

Once the optimal subset $\mathbf{y}$ is obtained, tracks are confirmed up to $t + \tau_{\text{step}}$:

$$\Gamma_{1:t+\tau_{\text{step}}} = \left\{\gamma^i_{1:t+\tau_{\text{step}}} \mid \gamma^i_{1:\tau_{\text{depth}}} \in H_t(\tau_{\text{depth}}),\ \mathbf{y}^i = 1\right\}.$$

Instead of regenerating hypotheses from scratch from $H_{t+\tau_{\text{step}}}(0)$, we exploit previously constructed hypotheses with $H_{t+\tau_{\text{step}}}(\tau_{\text{depth}} - \tau_{\text{step}}) \subset H_t(\tau_{\text{depth}})$, so only incompatible branches are pruned to directly build $H_{t+\tau_{\text{step}}}(\tau_{\text{depth}} - \tau_{\text{step}})$.

To further reduce complexity, hypotheses are early pruned under two conditions:

1. **Termination:** When continuing a missed detected hypothesis γ is less probable than declaring its termination on its last detected frame and starting a new track from any potential detection, we stop considering associations. The condition writes:

$$G^{\text{d}}(\gamma) > G^{\text{c}}(\gamma)\frac{(1-p_{\text{d}})(1-p_{\text{m}})}{\lambda_{\text{b}}}\frac{A}{B^{\text{b}}}$$

   where A upper bounds $p(\cdot \mid Z^\gamma)$ and B^{b} lower bounds $p(\cdot)$.

2. **False positive:** When continuing a missed detected hypothesis γ is less probable than classifying its associated detections as false positive and starting a new track from any potential detection, we stop considering associations. This condition writes:

$$G^{\text{f}}(\gamma) > G^{\text{c}}(\gamma)\frac{(1-p_{\text{d}})(1-p_{\text{m}})}{\lambda_{\text{b}}}\frac{A}{B^{\text{f}}}$$

   where B^{f} is a lower bound of $p^{\text{f}}(\cdot)$.

This pruning strategy reduces the computational load without compromising the Bayesian optimality of the resulting solution. Indeed, the pruned hypotheses would not have been selected

Table V.2: Summary of MHT parameters, their interpretation, and proposed estimation methods.

Parameter	Meaning	Estimation method
p_{d}	Target termination probability	Manual estimate (disappearance rate)
p_{b}	Target birth rate	Manual (growth rate) or assume $p_{\mathrm{b}} = p_{\mathrm{d}}$
p_{m}	Miss-detection probability	Detector FNR, or manual/annotation-based
p_{f}	False positive probability	Detector FPR, or manual/annotation-based
$\lambda_{\mathrm{f}}(t)$	Mean # false positives at t	$\hat{\lambda}_{\mathrm{f}}(t) = p_{\mathrm{f}} M_t$
$\lambda_{\mathrm{b}}(t)$	Mean # new births at t	$\tilde{\lambda}_{\mathrm{b}}(t)$ (Eq. V.36)
N_0	Initial unseen targets	Estimated from M_1 (Eq. V.35)
V	Measurement volume	Image volume or user-defined
η	Association threshold	Defaults to a minimal value (Eq. V.37)

regardless, as they are dominated by a more probable concurrent hypothesis (either death + birth or false positive + birth).

Finally, associations hypotheses are only generated for likely measurements, such that

$$p(\cdot \mid Z^{\gamma}) \geq \eta, \tag{V.26}$$

where the likelihood threshold η limits tree growth. By default, η is chosen to prune as many hypotheses as possible while keeping the Bayesian optimality of the solution (see Chapter V.2.4).

Even with our efficient pruning, the number of hypotheses follows roughly $n_t = (1+k)n_{t-1} + M_t$, with k expected feasible detections per hypothesis ($k \ll 10$ in practice). Thus τ_{depth} must remain small (typically < 10).

V.2.4 Parameter Estimation

Bayesian MHT trackers rely on several key parameters that must be estimated prior to execution in order to achieve reliable tracking performance. In this section, we propose simple estimators that require only coarse prior knowledge about the targets and the detector. This is summarized in Table V.2. We acknowledge that the estimators developed here may not be perfectly suited to all tracking scenarios; however, they provide a reasonable approximation for a wide range of general tracking problems. Extensions or refinements can be introduced as needed to address specific challenges.

We denote by $N_t < N$ the true number of active targets at frame t, and N_0 the number of active targets before the first frame. According to our model, each active target can terminate with constant probability p_{d}, and each surviving target can be miss-detected with constant probability p_{m}. In addition, the number of new targets M_t^{b} and false positives M_t^{f} are modeled as Poisson random variables:

$$M_t^{\mathrm{b}} \sim \mathcal{P}(\lambda_{\mathrm{b}}(t)), \qquad M_t^{\mathrm{f}} \sim \mathcal{P}(\lambda_{\mathrm{f}}(t)).$$

Note that λ_{b} and λ_{f} are time dependent.

Termination and missed detections. The termination probability p_d can usually be estimated manually by inspecting the proportion of targets that disappear between consecutive frames (*e.g.*, leaving the field of view or shutting off). The missed-detection probability p_m corresponds to the false negative rate of the detector. We assume it is constant across targets and frames. It is typically known for a given detection method, but can also be estimated manually or with a small set of annotated detections.

False positives. To simplify estimation of $\lambda_\mathrm{f}(t)$, we introduce p_f as the detector's false positive rate, that we assume to be detection and time-invariant. It relates to λ_f via

$$\lambda_\mathrm{f} = \mathbb{E}(M_t^\mathrm{f}) = \mathbb{E}\Big(\mathbb{E}(M_t^\mathrm{f} \mid M_t)\Big) = \mathbb{E}(p_\mathrm{f}\, M_t) = p_\mathrm{f}\, \mathbb{E}(M_t). \tag{V.27}$$

Thus, given p_f, we propose the unbiased single-frame estimator:

$$\hat{\lambda}_\mathrm{f}(t) = p_\mathrm{f}\, M_t. \tag{V.28}$$

We note that aggregating several frames could reduce variance of the $\mathbb{E}(M_t)$ estimations. But this would introduce additional parameter dependencies, therefore, we decided to rely on this simple online estimator, which we found satisfactory in practice.

Births. Similarly, we introduce p_b, the proportional birth rate, defined such that

$$\mathbb{E}(M_t^\mathrm{b} \mid N_t) = p_\mathrm{b} N_t, \qquad \lambda_\mathrm{b}(t) = p_\mathrm{b}\mathbb{E}(N_t).$$

The target population evolves as

$$\mathbb{E}(N_t) = (1 - p_\mathrm{d})\mathbb{E}(N_{t-1}) + \lambda_\mathrm{b}(t), \tag{V.29}$$

$$(1 - p_\mathrm{b})\mathbb{E}(N_t) = (1 - p_\mathrm{d})\mathbb{E}(N_{t-1}). \tag{V.30}$$

In particular, for a stationary number of targets, $p_\mathrm{b} = p_\mathrm{d}$. Otherwise, p_b can be estimated from the rough growth rate of the target population. Given p_b, an unbiased single-frame estimator of $\lambda_\mathrm{b}(t)$ follows from:

$$\mathbb{E}(M_t) = (1 - p_\mathrm{d})(1 - p_\mathrm{m})\mathbb{E}(N_{t-1}) + \lambda_\mathrm{b}(t) + \lambda_\mathrm{f}(t), \tag{V.31}$$

$$\mathbb{E}(N_t) = \frac{1 - p_\mathrm{f}}{(1 - p_\mathrm{b})(1 - p_\mathrm{m}) + p_\mathrm{b}}\mathbb{E}(M_t), \tag{V.32}$$

$$\hat{\lambda}_\mathrm{b}(t) = \frac{p_\mathrm{b}(1 - p_\mathrm{f})}{(1 - p_\mathrm{b})(1 - p_\mathrm{m}) + p_\mathrm{b}} M_t. \tag{V.33}$$

Correction for initial targets. Initial unseen targets (N_0) produce additional pseudo-births in early frames. Let M_t^0 denote the number of such detections at frame t. Its expectation is

$$\mathbb{E}(M_t^0) = (1 - p_\mathrm{d})^t (p_\mathrm{m})^{t-1}(1 - p_\mathrm{m})\mathbb{E}(N_0). \tag{V.34}$$

An estimator of N_0 is obtained from M_1:

$$\hat{N_0} = \frac{1 - p_\mathrm{f}}{(1 - p_\mathrm{b})(1 - p_\mathrm{m}) + p_\mathrm{b}} \cdot \frac{1 - p_\mathrm{b}}{1 - p_\mathrm{d}} M_1. \tag{V.35}$$

Hence, we propose a corrected estimator for $\lambda_\mathrm{b}(t)$:

$$\tilde{\lambda}_\mathrm{b}(t) = \hat{\lambda}_\mathrm{b}(t) + (1 - p_\mathrm{d})^t (p_\mathrm{m})^{t-1}(1 - p_\mathrm{m})\hat{N}_0, \tag{V.36}$$

which only impacts early frames, as the correction quickly vanishes as $p_\mathrm{m}^{t-1} \ll 1$.

Birth and false positive densities. For new targets or false positives, the measurement $\mathbf{z}_t^j$ is assumed uniformly distributed in the measurement space:

$$p(\mathbf{z}_t^j) = p^{\mathrm{f}}(\mathbf{z}_t^j) = \frac{1}{V}.$$

where V is the volume of the measurement space (usually the volume of the images). In practice, V can be adjusted manually if targets occupy only a subregion of the image.

Association densities. The likelihood of linking a detection $\mathbf{z}_t^j$ to a track $\gamma_{1:t-1}^i$ is $p(\mathbf{z}_t^j \mid Z^{\gamma_{1:t-1}^i})$, computed using different state-space models detailed in Chapter V.2.5.

Association threshold. Finally, the likelihood threshold η should be the largest value that prunes invalid hypotheses without discarding correct ones. We set η based on the following comparison: given a track $\gamma_{1:t-1}^i$ and detection $\mathbf{z}_t^j$, there are three scenarios producing a single resulting track: (1) link $\gamma_{1:t-1}^i$ to $\mathbf{z}_t^j$; (2) miss the track and classify $\mathbf{z}_t^j$ as false positive; (3) terminate the track and start a new track from $\mathbf{z}_t^j$. Scenario (1) is necessarily rejected if its likelihood is lower than both alternatives:

$$p(\mathbf{z}_t^j \mid Z^{\gamma_{1:t-1}^i}) \leq \min\left(\frac{p_{\mathrm{d}}\lambda_{\mathrm{b}}(t)}{(1-p_{\mathrm{d}})(1-p_{\mathrm{m}})V},\ \frac{p_{\mathrm{m}}\lambda_{\mathrm{f}}(t)}{(1-p_{\mathrm{m}})V}\right). \tag{V.37}$$

We therefore adopt this value as the default η, since linking hypotheses with smaller likelihoods would not be selected in any cases. Stricter thresholds may be set manually to further reduce complexity.

Summary. The remaining four parameters that require user specification are:

- p_{m}: probability that a target is missed (detector false negative rate),
- p_{f}: probability that a detection is false positive (detector false positive rate),
- p_{d}: probability that a target terminates (estimated from disappearing targets),
- p_{b}: birth rate of new targets (equal to p_{d} or estimated from appearing targets).

All other parameters are either derived through the estimators above or are introduced by the association densities (see Chapter V.2.5).

These parameters and estimators rely on the following three assumptions which, while not fully realistic, provide a reasonable approximation for most biological tracking scenarios.

1. **Target termination.** Each active target has, at every frame t, a constant probability of disappearing p_{d}, independent of both time and spatial location.
2. **Target birth.** At each frame t, a constant rate of target birth p_{b} is assumed, also independent of time and location. This leads to an expected number of births $\lambda_{\mathrm{b}}(t) = p_{\mathrm{b}}\mathbb{E}(N_t)$, proportional to the number of active targets N_t. We argue that this assumption is more realistic than using a fixed expected number of births λ_{b} [26], as it accounts for variations in target density or cell mitosis events, which naturally scale with the number of existing cells.

3. **Detector performance.** The detector is assumed to have time and target-invariant performance, characterized by a false negative rate p_{m} and a false positive rate p_{f}.

We acknowledge that these assumptions could be refined in future work — for example, by incorporating spatially dependent birth and death processes (e.g., higher rates near image borders) or temporally correlated false negatives (a missed detection at frame t suggesting an increased likelihood of another at t+1 for the same target, as modeled in EMHT [26]).

V.2.5 State-Space Filtering

To model target motion and sensor noises within MHT, we rely on Kalman filtering as introduced in Chapter IV. We refer the reader to Chapter IV.2.1 for the description of these models. The detection process provides noisy position measurements $\mathbf{z}_t^{\mathrm{pos}}$, while optical flow can supply noisy velocity measurements $\mathbf{z}_t^{\mathrm{vel}}$. We denote by

$$\mathbf{z}_t = \begin{pmatrix} \mathbf{z}_t^{\mathrm{pos}} \\ \mathbf{z}_t^{\mathrm{vel}} \end{pmatrix}$$

the concatenated measurement vector. For clarity, we omit track indices i and detection indices j in this section.

Kalman filtering enables recursive estimation of the predictive likelihood of the next measurement $p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}) = p(\mathbf{z}_t^{\mathrm{pos}}|\mathbf{z}_{1:t-1}) \cdot p(\mathbf{z}_t^{\mathrm{vel}} \mid \mathbf{z}_t^{\mathrm{pos}}, \mathbf{z}_{1:t-1})$ (see Chapter IV). In practice, we consider both standard Kalman filters and optical-flow-enhanced Kalman filters, with motion dynamics modeled either as Brownian or directed motion (see Chapter IV.2.3).

As in the previous chapter, we restrict the likelihood to positional measurements $p(\mathbf{z}_t^{\mathrm{pos}}|\mathbf{z}_{1:t-1})$, rather than using the full joint likelihood of position and (future) velocity $p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1})$. This choice is motivated by three practical considerations:

1. **Limited discriminative power of the future velocity.** Due to the spatial regularization inherent in optical flow, neighboring targets often share similar velocity measurements. As a result, the velocity measurement $\mathbf{z}_t^{\mathrm{vel}}$ (between frame t and $t+1$) adds little information for resolving identity ambiguities during data association. We emphasize that in KOFT and KOFT-MHT, the velocity measurements are already used to update the Kalman filter, enabling better positional predictions. Therefore, the positional likelihood $p(\mathbf{z}_t^{\mathrm{pos}} \mid \mathbf{z}_{1:t-1})$ in the data association between frame $t-1$ and t already accounts for the measured optical flow $\mathbf{z}_{t-1}^{\mathrm{vel}}$ between frame $t-1$ and t.
2. **Temporal delay.** The velocity measurement $\mathbf{z}_{t+\tau_{\mathrm{depth}}}^{\mathrm{vel}}$ becomes available only once frame $t+\tau_{\mathrm{depth}}+1$ is acquired, adding an additional temporal delay of 1 frame. Our dual-update formulation allows this information to be incorporated at the next prediction time after frame $t+\tau_{\mathrm{depth}}+1$ is acquired, but the associations up to frame $t+\tau_{\mathrm{depth}}$ must be resolved beforehand.
3. **Consistency with priors.** Incorporating velocity into the likelihood would require extending both the birth likelihood $p(\mathbf{z}_t)$ and the false-positive likelihood $p^{\mathrm{f}}(\mathbf{z}_t)$ to include velocity components, typically modeled as uniform distributions. This introduces additional parameters without significantly improving association performance.

For these reasons, we neglect the velocity likelihood $p(\mathbf{z}_t^{\mathrm{vel}} \mid \mathbf{z}_t^{\mathrm{pos}}, \mathbf{z}_{1:t-1})$ and retain the positional one as the sole likelihood factor in the association stage of our MHT.

V.2.5.1 Filtering parameters

These models depend on three key additional parameters:

1. σ_{pos}: standard deviation (in pixels) of the positional measurement noise from the detector. Can be roughly estimated as half the radius of targets.

2. σ_{vel}: standard deviation (in pixels) of the velocity measurement noise from optical flow (unused in standard Kalman filters). It should be adapted to match the confidence on the optical flow precision on the data.

3. σ_{mov}: standard deviation (in pixels) of the process noise. For Brownian motion it reflects the expected displacement per frame, while for directed motion it quantifies the expected change in velocity between consecutive frames.

V.2.5.2 Initialization

When a new track is initialized from a detection $\mathbf{z}_t$, the initial Kalman state $\mathbf{x}_t \mid \mathbf{z}_t$ is modeled as

$$\mathbf{x}_t \mid \mathbf{z}_t \sim \mathcal{N}\left(\begin{pmatrix}\mathbf{z}_t \\ \mathbf{0}_d\end{pmatrix}, \begin{pmatrix}\sigma_{\text{pos}}^2\mathbf{I}_d & \mathbf{0}_d \\ \mathbf{0}_d & (k\sigma_{\text{mov}})^2\mathbf{I}_d\end{pmatrix}\right), \tag{V.38}$$

where the initial position is set to the observed detection with variance σ_{pos}^2, and the initial velocity is assumed to be zero with variance $(k\sigma_{\text{mov}})^2$. The constant k is set to 1 for Brownian motion and 3 for directed motion, reflecting the larger uncertainty in velocity under the directed model.

V.2.5.3 Covariance Growth

Under time-invariant noise assumptions, the Kalman filter covariance converges to a steady-state positional covariance $\bar{\mathbf{S}}_{\infty}^{\text{pos}}$ determined by the DARE Riccati equations [184]. However, during initialization or after consecutive missed detections, the uncertainty about the predicted position can grow substantially. If this is expected for the filter to work properly, it inflates the number of candidate hypotheses to explore.

The rate of covariance growth depends strongly on the Kalman filter used:

- **Brownian Kalman filter:** the positional variance grows linearly with the number of missed frames τ:
$$\bar{\mathbf{S}}_{t+\tau}^{\text{pos}} = \bar{\mathbf{S}}_{t}^{\text{pos}} + \tau\,\sigma_{\text{mov}}^2,$$

- **Directed Kalman filter:** the positional variance grows quadratically with τ (as it grows linearly on the velocity):
$$\bar{\mathbf{S}}_{t+\tau}^{\text{pos}} = \bar{\mathbf{S}}_{t}^{\text{pos}} + \tfrac{\tau(\tau+1)}{2}\sigma_{\text{mov}}^2.$$

- **Optical flow enhanced filters:** In both Brownian and Directed models, the velocity is still measured through optical flow when detections are missing. Therefore, the velocity uncertainty stays bounded by σ_{vel}^2. We found that the uncertainty on the position grows linearly following:
$$\bar{\mathbf{S}}_{t+\tau}^{\text{pos}} \leq \bar{\mathbf{S}}_{t}^{\text{pos}} + \tau\,\sigma_{\text{vel}}^2,$$

In practice, the quadratic growth in Directed Kalman filters can yield uncertainties more than an order of magnitude larger than the steady-state variance, leading to an exponential growth of candidate associations. To mitigate this effect in Directed Kalman Filter, we propose to restrict candidate hypotheses by discarding detections lying outside a radius of three standard deviations of $\|\bar{\mathbf{S}}_\infty^{\text{pos}}\|_\infty$ (Euclidean thresholding).

With our optical flow enhanced filters, such ad-hoc pruning is unnecessary: velocity updates stabilize the predictions and reduce uncertainty. Moreover, the steady-state variance $\bar{\mathbf{S}}_\infty^{\text{pos}}$ is significantly smaller than in standard models, leading to far fewer candidate hypotheses, faster computations, and the ability to explore larger look-ahead depths in MHT (see Chapter V.3.3).

V.2.6 State-Space Smoothing

The proposed MHT framework solves for the maximum a posteriori (MAP) associations $\Gamma_{1:T}$ (Eq. V.4). However, the ultimate goal of tracking is to recover accurate target states—primarily positions and velocities—along the estimated tracks. As discussed in Chapter V.2, this requires a final state estimation step given the associations (Eq. V.5).

Most existing methods [27,128], including Bayesian approaches [26,132], bypass this problem by directly using the noisy measured positions:

$$\mathbf{x}_t^i = \begin{cases} \mathbf{z}_t^j & \text{if } \gamma_t^i = \{j\}, \\ \text{NaN} & \text{otherwise,} \end{cases} \tag{V.39}$$

where NaN encodes the absence of a detection. Missing states are typically filled by interpolation, or by reusing the filtering estimate $\hat{\mathbf{x}}_t^i$ computed during association (often referred to as a *virtual detection*) [26].

In contrast, we propose a fully Bayesian state estimation step that leverages the same state-space models used for filtering. Assuming conditional independence between targets given the associations, the posterior distribution over states factorizes as:

$$\mathbb{P}(X_{1:T} \mid \Gamma_{1:T}, Z_{1:T}) = \prod_{i=1}^{N} \mathbb{P}(X_{1:T}^i \mid Z^{\gamma_{1:T}^i}). \tag{V.40}$$

State estimation can then be resolved independently for each target, as a single-target smoothing problem:

$$X_{1:T}^{\star,i} = \arg\max \mathbb{P}(X_{1:T}^i \mid Z^{\gamma_{1:T}^i}). \tag{V.41}$$

In our linear Gaussian case, the marginal distributions $\mathbb{P}(\mathbf{x}_t^i \mid Z^{\gamma_{1:T}^i})$ remain Gaussian and can be computed efficiently using the Rauch–Tung–Striebel (RTS) smoother (see Chapter II.3.1.6). The joint trajectory distribution across time is therefore also Gaussian, and its mean $(\hat{\mathbf{x}}_t^{\text{s,i}})_{1:T}$ provides the MAP estimate for $X_{1:T}^i$. Importantly, our optical flow enhanced Kalman smoother differ only in the filtering step, where it benefits from its two sensors, whereas the smoothing step itself remains identical.

This post-processing stage reduces positional errors and improves downstream analyses based on estimated positions or velocities. We note, however, that it only addresses positional uncertainties, and it does not impact the associations.

V.3 Experiments and Results

V.3.1 Tracking Setup

We evaluate our method on synthetic datasets generated by the SINETRA simulator (see Chapter III). We focus on 2D scenarios, where each synthetic video consists of $T = 200$ frames with a resolution of 1024×1024 pixels, and approximately $N \simeq 800$ objects to track. For each experiment, five random videos are generated using the same parameters but different random seeds. In tables, we report mean and standard deviation across the five runs, while in figures we display the 99.7% confidence interval.

To measure the accuracy of computed tracks, we use the Higher Order Tracking Accuracy (HOTA) metric [180], which accounts for localization, detection, and association performance (Chapter III). We use a matching tolerance of r pixels, meaning that a predicted track is considered correctly associated with a ground-truth object only if its position lies within r pixels of the ground truth. In practice, we set $r = 2.0$ pixels, corresponding to the typical size of the targets. Tracks deviating beyond this distance are considered biologically erroneous as they may compromise subsequent analyses.

We compare our SKT-MHT tracker, which uses a standard Kalman filter, and KOFT-MHT, which uses our optical flow-enhanced Kalman filter, with their frame-by-frame maximum likelihood counterparts (SKT and KOFT) developed in Chapter IV. To validate our Bayesian MHT framework, we additionally benchmark against two similar Track-Oriented MHT implementations for biological targets: the enhanced MHT (EMHT) [26] provided in the Icy software [50], and the MAP method [132] originally implemented in Matlab [185].

Both EMHT and MAP formulate the problem as a multi-dimensional assignment solved via mixed-integer linear programming, similar to our own approach. However, they differ in the modeling of costs, priors, and heuristics. MAP employs an Interactive Multiple Model (IMM) [57] to compute association likelihoods $p(\mathbf{z}_t^j \mid Z^{\gamma_{1:t-1}^i})$ and replaces prior probabilities with ad hoc penalty terms. As the original implementation is not maintained, we minimally updated the code to run on Matlab/Octave under Linux and wrapped it for Python. Since SINETRA does not generate split or merge events, we disabled this functionality in MAP, as it otherwise incorrectly interprets missed detections as merges. We set the maximum gap parameter to $\tau_{\max} = 5$ frames and remove tracks shorter than $\tau_{\min} = 2$ frames, since false positives are not explicitly modeled.

EMHT uses a similar IMM for association likelihoods, but is closer to our method for the Bayesian priors. It uses a Poisson prior for births and false positives. Instead of Bernoulli track termination, it models track lifetimes with a Poisson distribution, coupled with an existence probability threshold for automatic track termination and validation. The algorithm is actively maintained in the Icy software, which we wrapped for Python integration.

All methods rely on pre-detected objects. To isolate the contribution of the tracking algorithm, we re-used our *fake* detector (see Chapter IV) that generates detections from ground-truth positions with Gaussian positional noise ($\sigma_{\text{pos}} = 0.5$ pixels), and user-controlled false negative (FNR) and false positive (FPR) rates. The detector is denoted Fake@F1, with FPR = FNR $= 1 - \text{F1}$. As our *fake* detector does not produce detection merges or splits, we also added comparisons with a real detector. We used the wavelet-denoising and thresholding method (Wavelet spot detector) [24], which achieves about 85% recall and precision in our synthetic dataset.

Table V.3: Summary of experimental settings for all evaluated methods.

Method	Motion Model	Process Noise	Gap Handling	Threshold	Birth/Death Model	False Positives
SKT (ours)	Brownian / Directed	$\sigma_{\text{mov}} = 4.0$ / 2.0 px	Fixed gap $\tau_{\text{max}} = 7$	Likelihood ($3 \cdot 10^{-4}$)	None	Small tracks ($\tau_{\text{min}} = 3$)
KOFT (ours)	Brownian / Directed + Optical Flow	$\sigma_{\text{mov}} = 4.0$ / 2.0 px	Fixed gap $\tau_{\text{max}} = 7$	Likelihood ($3 \cdot 10^{-4}$)	None	Small tracks ($\tau_{\text{min}} = 3$)
SKT-MHT (ours)	Brownian / Directed	$\sigma_{\text{mov}} = 4.0$ / 2.0 px	Bayesian	Bayesian + Euclidean ($\sim$ 10px)	Poisson + Bernoulli	Poisson
KOFT-MHT (ours)	Brownian / Directed + Optical Flow	$\sigma_{\text{mov}} = 4.0$ / 2.0 px	Bayesian	Bayesian (automatic)	Poisson + Bernoulli	Poisson
EMHT [26]	IMM (Brownian + Directed)	Similar	Bayesian	Mahalanobis (4.0)	Poisson + Poisson track length	Poisson
MAP [132]	IMM (Brownian + Directed)	Similar	Fixed gap $\tau_{\text{max}} = 5$	Euclidean (7 px)	Ad hoc penalties	Small tracks ($\tau_{\text{min}} = 2$)

Since HOTA is sensitive to localization accuracy, all methods are evaluated after state-space smoothing (see Chapter V.2.6), increasing HOTA performance for all the methods that did not include this smoothing step, but allowing a fair comparison between approaches.

V.3.1.1 Parameters

Each tracker relies on a different motion model, but we harmonize the settings wherever possible. For Brownian motion, we set the process noise to $\sigma_{\text{mov}} = 4.0$ pixels, while for directed motion we set $\sigma_{\text{mov}} = 2.0$ pixels. This choice reflects SINETRA's dynamics: accelerations are typically below 6 pixels per frame, while velocities can reach up to 12 pixels per frame.

In SINETRA, the number of targets is constant, except for rare events where targets enter or leave the field of view. We therefore set the birth and death probabilities to $p_{\text{d}} = p_{\text{b}} = 10^{-4}$, corresponding to an expected one birth/death every 10 frames. For EMHT, the expected number of birth is set accordingly (plus a correction for pseudo-births on the first frame). In MAP, no such parameter exists.

For all Bayesian approaches, miss-detection and false-positive probabilities are set to match the true detector rates. We use an approximate 10% error rate for the Wavelet spot detector. We study the impact of erroneous estimation of these probabilities in Chapter V.3.4. MAP uses an Euclidean distance threshold of 7 pixels, while EMHT employs a Mahalanobis distance threshold of 4.0. These thresholds were selected to approximately match our Euclidean thresholding for the Directed Kalman filter (see Chapter V.2.5), with minor adjustments obtained through a small grid search to account for method-specific differences.

For MHT-based methods, we use a tree depth of $\tau_{\text{depth}} = 3$, corresponding to an effective look-ahead of two frames beyond frame-by-frame trackers. The step size is fixed at $\tau_{\text{step}} = 1$. We further explore the impact of tree depth in Chapter V.3.3.

Finally, for SKT and KOFT, we reuse the parameter settings established in Chapter IV. Table V.3 summarizes all the experimental settings of all the evaluated methods.

V.3.2 Tracking Results

Table V.4 reports the HOTA metric across all methods on SINETRA dataset [65] using Wavelet based spot detections [24]. Our Bayesian MHT algorithms (SKT-MHT and KOFT-MHT) consistently outperform their frame-by-frame counterparts (SKT and KOFT). For example, in Springs 2D, KOFT-MHT reaches 98.2% HOTA compared to 95.6% for KOFT; in Hydra-Flow, it achieves 96.1% versus 91.6%, respectively. Similar trends are observed for SKT-MHT, which improves upon SKT by more than 15% HOTA in Hydra-Flow.

Among MHT methods, MAP is outperformed in both scenarios, lagging behind fully Bayesian approaches. EMHT, on the other hand, achieves competitive results, particularly in Hydra-Flow where it approaches the performance of our SKT-MHT. We note that the improvement

Table V.4: Tracking performance (HOTA at 2 pixels tolerance [180]) on synthetic videos. Frame per second (FPS) are computed on the same laptop and do not account for the detection process.

Scenario	Springs 2D	Hydra Flow	FPS
MAP [132]	$79.1 \pm 4.8\%$	$63.9 \pm 6.9\%$	$\frac{1}{7}$
EMHT [26]	$86.9 \pm 3.7\%$	$80.7 \pm 7.3\%$	2
SKT (ours)	$82.3 \pm 4.4\%$	$65.9 \pm 5.7\%$	25
SKT-MHT (ours)	$88.3 \pm 2.2\%$	$81.0 \pm 4.8\%$	5
KOFT (ours)	$95.6 \pm 0.8\%$	$91.6 \pm 2.3\%$	10
KOFT-MHT (ours)	$\mathbf{98.2} \pm 0.3\%$	$\mathbf{96.1} \pm 1.4\%$	5

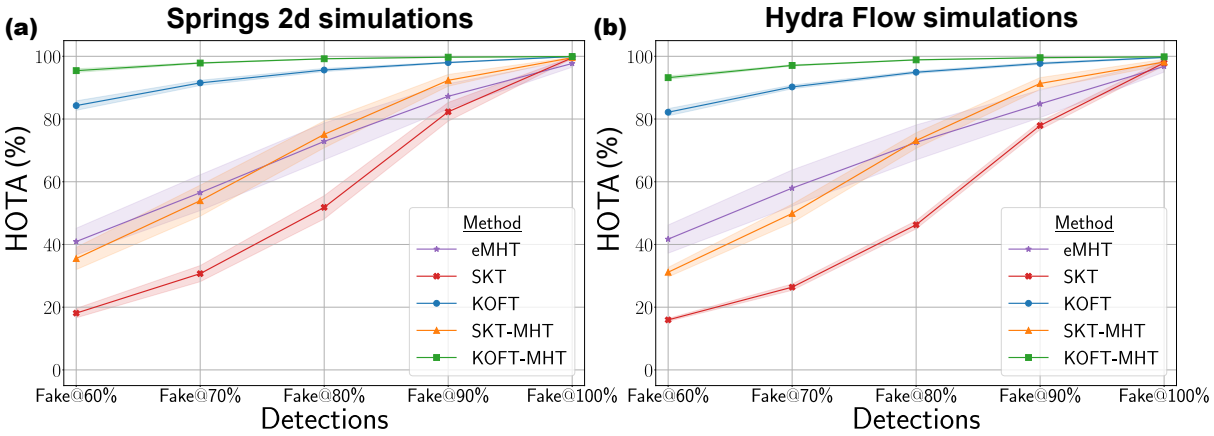


Figure V.3: Tracking performance on synthetic videos. The HOTA metric [180] with 2 pixels tolerance is reported with respect to detection accuracy (Fake@F1). The light-colored envelope around each curve corresponds to the 99.7% confidence interval over $M = 5$ simulations. **(a)** Results on Springs 2D synthetic videos **(b)** Results on Hydra Flow synthetic videos

of EMHT compared to earlier chapters is due to the inclusion of our smoothing post-processing step for fair comparison. Overall, KOFT-MHT provides the most accurate results, reducing the gap to perfect tracking (100% HOTA) by more than half relative to KOFT. For instance, in Springs 2D, KOFT still makes 4.4% errors, whereas KOFT-MHT reduces this to only 1.8%.

Computational efficiency. Runtime comparisons are also reported in Table V.4. MAP is by far the slowest, requiring approximately 7 seconds per frame. EMHT runs at 2 FPS in our setup. Our Bayesian MHT implementations are about twice as fast as EMHT, owing to an optimized implementation and efficient pruning strategies. Notably, KOFT-MHT runs at the same speed as SKT-MHT (5 FPS), despite the additional optical flow computation. This is because optical flow reduces the number of competing hypotheses, thereby accelerating MHT optimization. The benefits become more pronounced at larger tree depths, as analyzed in Chapter V.3.3.

Impact of detection quality. To further assess robustness, we replaced the wavelet detector with the artificial *fake* detector (see Chapter IV.3.3.1), allowing controlled variation of false positive and false negative rates (Figure V.3). Due to its prohibitive runtime, MAP was excluded from these experiments. Results show that our MHT solutions are substantially more

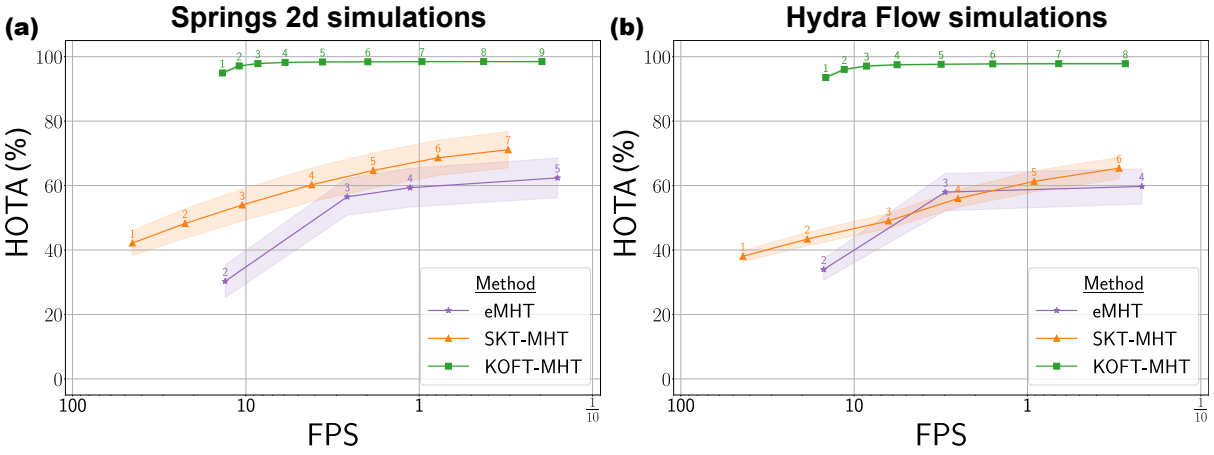


Figure V.4: Evolution of tracking performance and runtime with respect to the tree depth on synthetic videos. The HOTA metric [180] with 2 pixels tolerance is reported against FPS at different tree depths (annotated on each curve). The same Fake@70% detector is used across all experiments. The light-colored envelope around each curve corresponds to the 99.7% confidence interval over M = 5 simulations. **(a)** Results on Springs 2D synthetic videos **(b)** Results on Hydra Flow synthetic videos

robust to detection errors. At low detection quality, EMHT outperforms SKT-MHT, owing to its IMM motion model. However, Appendix A.2 demonstrates that when SKT-MHT is equipped with a Brownian motion model, it outperforms EMHT in this regime. Most importantly, KOFT-MHT consistently maintains above 90% HOTA across all detection settings, clearly outperforming all other approaches.

V.3.3 Depth impact

We next investigated the impact of the tree depth parameter τ_{depth} on tracking performance. To avoid an early saturation of HOTA scores for KOFT-MHT, we relied on a degraded Fake@70% detector with around 30% false positives and 30% missed detections. Figure V.4 reports HOTA scores [180] (2-pixel tolerance) against frame-per-second (FPS) runtimes for different tree depths ($\tau_{\text{depth}} = 1, \ldots, 9$). Experiments exceeding 30 minutes of runtime were terminated, and the corresponding tree depth was omitted.

First, we observed that all methods were consistently slower on the Hydra Flow scenario than on the Springs 2D scenario. For example, at $\tau_{\text{depth}} = 3$, SKT-MHT runs about twice as slowly on Hydra Flow as on Springs 2D. This runtime drop mirrors the decrease in accuracy: in Hydra Flow, targets are slightly closer, which increases the number of potential branches to explore at each depth.

Second, performance improved with increasing τ_{depth}, as expected for Bayesian trackers. On Springs 2D, SKT-MHT increased from about 40% HOTA at $\tau_{\text{depth}} = 1$ to 70% at $\tau_{\text{depth}} = 7$. Similarly on Hydra Flow, KOFT-MHT improved from 93.5% HOTA at $\tau_{\text{depth}} = 1$ to a plateau above 97.5% for $\tau_{\text{depth}} \geq 6$.

Compared with EMHT, our standard Bayesian tracker (SKT-MHT) is generally faster: for the same runtime budget, it can operate at a higher effective tree depth (+1 or +2) and thus achieve better performance. Moreover, SKT-MHT shows better tracking performance scaling, where EMHT seems to saturate quicker.

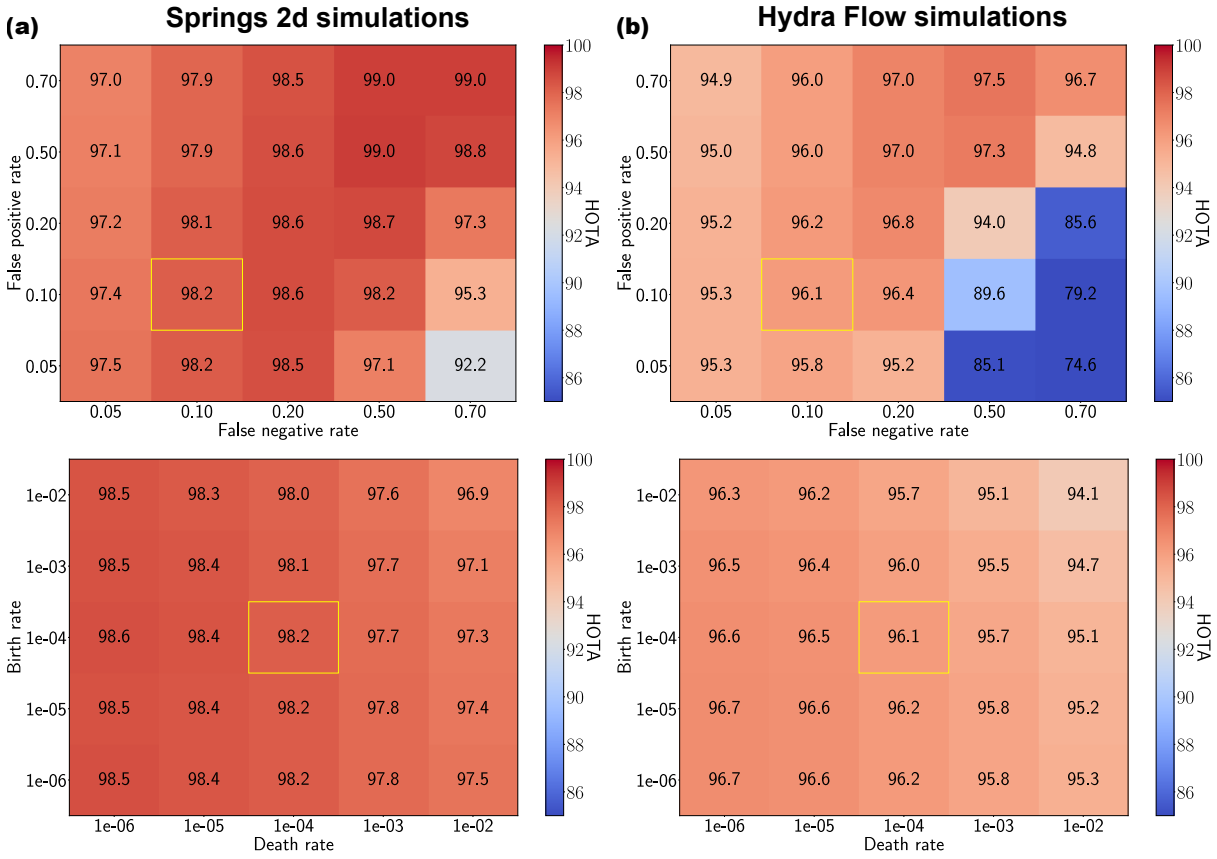


Figure V.5: Sensitivity of KOFT-MHT to its a priori parameters. Heatmap of the HOTA metric [180] with 2-pixel tolerance using the Wavelet spot detector [24]. The default configuration is highlighted in yellow, with variations in the expected false positive and negative rates of the detector (top), or in the birth and death rates of targets (bottom). **(a)** Results on Springs 2D synthetic videos. **(b)** Results on Hydra Flow synthetic videos.

Finally, KOFT-MHT is slower than SKT-MHT at small depths due to the overhead of optical flow computation. However, at larger depths, the more precise motion model substantially prunes the hypothesis space, reducing the number of branches explored. As a result, KOFT-MHT not only achieves higher accuracy but also becomes faster than SKT-MHT at high tree depths.

These results highlight that deeper trees consistently benefit Bayesian trackers, and that KOFT-MHT scales more favorably with depth than other methods thanks to its stronger motion model.

V.3.4 Parameter Sensitivity

We analyzed the impact of our four key a-priori parameters required by our Bayesian trackers: (1) the false positive rate of the detector p_f, (2) the false negative rate of the detector p_m, (3) the birth rate of targets p_b, and (4) the death rate of targets p_d.

By default, we set $p_\mathrm{b} = p_\mathrm{d} = 10^{-4}$, since in SINETRA datasets birth and death events are rare, occurring only when targets enter or exit the field of view. For the Wavelet detector, we set $p_\mathrm{f} = p_\mathrm{m} = 0.1$, approximating the detector's empirical error rates.

Figure V.5 reports the performance when these default values are varied. First, we observe

that the chosen birth and death rates are slightly higher than optimal: smaller values (e.g., 10^{-6}) yield a modest performance gain of about +0.4% HOTA. More importantly, performance remains stable across several orders of magnitude: over- or under-estimating p_{b} and p_{d} by one to two orders of magnitude reduces accuracy by less than 1% HOTA.

Interestingly, using the detector's true false positive and negative rates appears suboptimal. Increasing the expected detection error rates to $p_{\mathrm{f}} = p_{\mathrm{m}} = 0.5$ consistently improves performance by more than 0.8% HOTA across both scenarios. The reasons behind this phenomenon remain unclear; without further theoretical justification or experimental validation, we recommend continuing to use the detector's expected error rates. Nevertheless, the results demonstrate robustness: even a twofold over- or under-estimation of p_{f} and p_{m} has only a limited impact on performance. However, we observe that overestimating the false negative rate p_{m} while underestimating the false positive rate p_{f} leads to a significant drop in performance. In this case, the linker becomes overly permissive, generating spurious tracks by avoiding track termination (since missed detections are deemed likely) and by failing to classify short tracks as false positives.

Overall, these experiments show that while the default parameterization is not strictly optimal on SINETRA datasets, it maintains high tracking performances. In practice, this stability is valuable, as it allows approximate manual estimates of the a-priori parameters without risking significant performance degradation.

V.4 Discussion

In this chapter, we introduced a novel Bayesian tracking framework that supports both standard Kalman filtering and our optical-flow-enhanced Kalman filter. The framework employs a Bernoulli prior for track death and missed detections, and a Poisson prior for track births and false positives. To facilitate practical use, we proposed simple estimators that reduce the full parametrization of the priors to only four manually estimable parameters: birth rate, death rate, false negative rate, and false positive rate. The implementation is based on a Track-Oriented Multiple Hypothesis Tracking (MHT) formulation, efficiently realized in Python with PyTorch and HiGHS.

Results on SINETRA synthetic datasets demonstrate that our standard Bayesian tracker (SKT-MHT) outperforms state-of-the-art MHT approaches using similar motion models, highlighting the robustness of our Bayesian framework. Thanks to its efficient implementation, SKT-MHT is also at least twice as fast as competing MHT methods. More importantly, our optical-flow enhanced Bayesian tracker (KOFT-MHT) combines the strengths of the KOFT motion model with the Bayesian framework, yielding the best overall performance. KOFT-MHT halves the number of errors made by KOFT, and—although optical flow initially increases computational load—it enables stronger hypothesis pruning, improving optimization efficiency. As a result, KOFT-MHT is even faster than SKT-MHT for look-ahead horizon larger than 3 frames, and is able to scale up to 10 frames depth.

We also observed that at relatively shallow tree depths ($\tau_{\mathrm{depth}} = 3$), the Brownian Kalman filter outperforms the directed Kalman filter of SKT-MHT under low detection quality. This suggests that equipping SKT-MHT with an Interactive Multiple Model (IMM) would improve robustness across diverse conditions. An IMM could also enhance KOFT-MHT, automatically balancing Brownian, Directed, and optical-flow-based motion models.

A remaining limitation of the current Bayesian framework is its inability to handle merged

and split detections, which frequently occur with unsupervised detectors. This limitation partly explains the lower performance achieved with the Wavelet spot detector compared to our controlled Fake@80% detector.

For our Hydra use case, KOFT-MHT substantially improves tracking accuracy, especially for neurons imaged with TdTomato fluorophores. However, for GCaMP imaging, the current Bayesian framework does not support neurons strong intermittent detectability, where a neuron can switch to a undetectable state for tens of frames. This limitation will be the focus of the following chapter.

Chapter VI

VEMC2: Tracklet Stitching with both Visual and Motion Cues

In this Chapter, we address the problem of long-range consecutive missed detections with a dedicated post-processing step, referred to as *Tracklet Stitching*. When a target is undetected for more than 10 consecutive frames, most MOT algorithms—including those developed in this thesis—fragment the trajectory into several shorter tracks, or *tracklets*, that each follow the same target over different temporal segments. We propose to improve this tracklet-stitching step by leveraging self-supervised contrastive visual features, enabling the recovery of full trajectories from fragmented ones.

This work has been published at the IEEE ISBI conference in 2023 as: Raphael Reme*, Victor Piriou, Alison Hanson, Rafael Yuste, Alasdair Newson, Elsa Angelini, Jean-Christophe Olivo-Marin, and Thibault Lagache. *Tracking intermittent particles with self-learned visual features.* IEEE ISBI, 2023. [67]

VI.1 Motivation

In the previous Chapters, we developed robust MOT algorithms for tracking Hydra neurons when they are detectable. In TRASE-IN, the dual-labeling of Hydra with TdTomato and GCaMP fluoropores [22], allowed to track neurons in the TdTomato channel which is independent of neural activity, and then extract the neuronal activity from the GCaMP channel. However, such dual-labeling is experimentally challenging, and in many cases only a single GCaMP channel is available. In GCaMP imaging, neurons exhibit intermittent detectability: they are visible only when active, and can remain undetectable for hundreds of frames. Similar issues also arise from imaging conditions (*e.g.*, low SNR, out-of-focus motion, or occlusions) and for other biological targets such as quantum dots [68] or photo-switchable fluorophores [69].

When targets are missed for more than 10 consecutive frames, most MOT algorithms, including those proposed in this thesis, fragment trajectories into multiple short *tracklets*. These must then be linked into full-length trajectories through a dedicated *tracklet stitching* post-processing step. Early approaches, such as [68], used local intensity to compute minimal paths between tracklets, but were designed for well-separated targets. More general approaches were later developed based on computing pairwise tracklet-to-tracklet costs and solving an optimal assignment with a Linear Assignment Problem (LAP) [27, 28], typically using position-based distances. Nonetheless, none of the above methods tried to exploit the visual signature of targets, which is usually preserved through time.

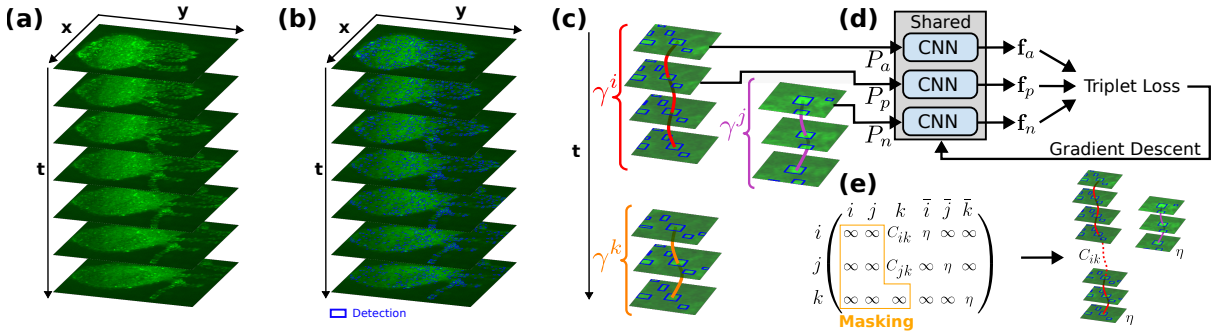


Figure VI.1: Overview of the complete tracking pipeline, with our proposed visual tracklet-stitching method for Hydra's neurons. **(a)** Input time-lapse sequence (GCaMP imaging). **(b)** Target detection in each frame, here using the Wavelet spot detector [24]. **(c)** Detections are linked into short trajectories (*tracklets*). At the time of this work (2022), the EMHT algorithm [26] was employed, as the more robust KOFT/BKOFT algorithms had not yet been developed. In this example, γ^i and γ^k track the same neuron at different temporal segments, whereas γ^j follows another neuron. **(d)** Self-supervised learning of discriminative visual features from patches around tracked targets. Triplet loss is used to produce discriminative target-specific features. **(e)** A hybrid tracklet-to-tracklet cost, combining visual and positional cues, is constructed. Temporal masking of backward links reduces computational load and improves stitching accuracy. A tracklet can remain unlinked by associating it to a virtual tracklet $(\bar{i}, \bar{j}, \bar{k})$ with cost η. The optimal assignment minimizes the total cost; on this simple example, the minimal cost is $C_{ik} + 2\eta$, linking γ^i to γ^k, while γ^j and γ^k remain unlinked.

In this Chapter, we extend the LAP framework for tracklet stitching by incorporating both positional and visual information into the tracklet-to-tracklet cost. To compute robust visual costs, we extract square image patches around detected targets and embed them into discriminative features using a self-supervised convolutional neural network (CNN). We demonstrate that these *visual features* allow for robust stitching of fragmented tracklets into accurate full-length trajectories, outperforming state-of-the-art approaches for Hydra's neurons.

VI.2 Method

VI.2.1 Tracklets

Our method takes as input the output of an arbitrary tracking algorithm. At the time of this work (2022), our own MOT algorithms had not yet been developed, and we therefore relied on existing state-of-the-art unsupervised tracking-by-detection approaches.

For the detection stage (Figure VI.1.a–b), we used the Wavelet spot detector [24], which only requires manual adjustment of the spot scale and detection threshold, and generally yields reliable detections. For the linking stage, we employed EMHT [26], implemented in the Icy software [50]. EMHT is a Bayesian Multiple Hypothesis Tracking algorithm (see Chapter II.1.3.4) that can effectively handle short-term (<5 consecutive frames) missed detections.

Formally, for a given video the tracking algorithm produces a set of N tracklets $\Gamma_{1:T} = \{\gamma^i_{1:T}\}_{1 \leq i \leq N}$. For each tracklet i at frame t, let $\mathbf{w}^i_t \in \mathbb{R}^d$ denote its pixel coordinates. We denote by $\mathbf{w}^i_{t^i_\mathrm{b}:t^i_\mathrm{d}}$ the sequence of positions between its birth at $t^i_\mathrm{b} \geq 1$ and its death at $t^i_\mathrm{d} \leq T$, where T is the total number of frames in the time-lapse sequence.

VI.2.2 Tracklet Stitching

As outlined above, the goal of tracklet stitching is to recover complete trajectories from fragmented tracklets (*e.g.*, γ^i and γ^k in Figure VI.1.c), which typically arise when targets are only intermittently detectable.

We formulate the stitching problem as a Linear Assignment Problem (LAP), following [27, 28]. The LAP seeks the optimal set of tracklet-to-tracklet *links* $\mathcal{L}$ that minimizes a linking cost $\mathbf{C}$. A link between two tracklets $\gamma^i, \gamma^j \in \Gamma_{1:T}$ is denoted as $(i \to j) \in \mathcal{L} \subset [\![1, N]\!]^2$. Note that a single trajectory may be fragmented into n tracklets, in which case $n-1$ links must be recovered. For example, if γ^i, γ^j, and γ^k correspond to the same target, the correct solution requires both $(i \to j)$ and $(j \to k)$ to be included in $\mathcal{L}$.

The main contribution of this work is the design of a hybrid tracklet-to-tracklet cost matrix $\mathbf{C} \in \mathbb{R}^{N \times N}$ that jointly leverages learned visual features and positional information. Optimal assignments $\mathcal{L}$ are then computed using the Jonker–Volgenant algorithm [30]. Because not all tracklets should be linked, the LAP is augmented with *virtual tracklets* (Figure VI.1.e) Linking a real tracklet to a virtual one incurs a fixed cost η, which represents the maximum acceptable linking cost. In practice, this ensures that a tracklet remains unlinked whenever all potential links exceed η.

Different sources of information (*e.g.*, position, appearance, motion) can contribute to defining tracklet similarity. However, the optimal combination is typically data-dependent. The following sections detail the construction of our hybrid cost matrix $\mathbf{C}$, which integrates both positional and visual cues to achieve robust stitching of fragmented trajectories.

VI.2.2.1 Positional cost

A simple way to define positional cost for linking two tracklets γ^i, γ^j is to take the distance between the last position of the first tracklet and the first position of the second tracklet [27]. Let's assume that $t^i_{\mathrm{d}} \leq t^j_{\mathrm{b}}$, then

$$\mathbf{C}^{\mathrm{pos}}_{ij} = \left|\left| \mathbf{w}^i_{t^i_{\mathrm{d}}} - \mathbf{w}^j_{t^j_{\mathrm{b}}} \right|\right|_2 \tag{VI.1}$$

with $||.||_2$ the Euclidean norm.

In some cases, target motion is spatially correlated. For example, when motion is induced by a common support, such as cells in tissues (*e.g.*, neurons in Hydra) or molecules of the cell membrane. Then, the motion of neighbor targets can provide information on the trajectory of a given target. With this in mind, the EMC2 algorithm [28] predicts the forward and backward positions of targets using Thin Plate Splines (TPS) interpolation between positions of tracked neighbors. Let $\hat{\mathbf{w}}^i_t$ be the forward TPS estimate of the target position after the tracklet termination, *i.e.* for frames $t \geq t^i_{\mathrm{d}}$. Similarly, for frames before tracklet birth, *i.e.* for frames $t \leq t^i_{\mathrm{b}}$, we define $\hat{\mathbf{w}}^i_t$ as the backward TPS estimate of the target position. Then, assuming that $t^i_{\mathrm{d}} \leq t^j_{\mathrm{b}}$, the positional cost defined in [28] is given by:

$$\mathbf{C}^{\mathrm{EMC2}}_{ij} = \min_{t \in [\![t^i_{\mathrm{d}}, t^j_{\mathrm{b}}]\!]} \left|\left| \hat{\mathbf{w}}^i_t - \hat{\mathbf{w}}^j_t \right|\right|_2 \tag{VI.2}$$

VI.2.2.2 Visual cost

In addition to positional information, visual features can also be used to compare and define a cost between tracklets. Let assume that each tracklet γ^i at each frame t can be characterized

by visual features $\mathbf{f}_t^i \in \mathbb{R}^p$, with p the dimension of the feature vector. These features are build to describe the visual content (shape, noise, texture) of a target and to discriminate between different targets (see Chapter VI.2.4). We can then define a visual cost for the link $(i \to j)$ as the smallest distance between the visual features of γ^i and γ^j:

$$\mathbf{C}_{ij}^{\text{vis}} = \min_{t^i \in [\![t_\text{b}^i, t_\text{d}^i]\!], t^j \in [\![t_\text{b}^j, t_\text{d}^j]\!]} \left|\left| \mathbf{f}_{t^i}^i - \mathbf{f}_{t^j}^j \right|\right|_2 \tag{VI.3}$$

VI.2.2.3 Temporal masking

To eliminate impossible or redundant links, we use temporal information, and define a boolean mask $\mathbf{M} \in \{0,1\}^{N \times N}$ as $\mathbf{M}_{ij} = \mathbb{1}(t_\text{d}^i \geq t_\text{b}^j)$. When $\mathbf{M}_{ij} = 1$, the cost calculation is skipped and $\mathbf{C}_{ij} = \infty$: no link can be made between tracklets γ^i and γ^j.

The mask simply states that the cost of $(i \to j)$ is computed only when γ^i ends before γ^j begins, allowing only forward temporal linking. The benefits are twofold: it improves computation time (less than half of the links are considered) and it also improves the overall performance of the stitching because it avoids competition between backward and forward temporal links.

Indeed, with a target tracked by three or more tracklets $(i \to j \to k)$, the link $(j \to i)$ is incompatible with $(j \to k)$ and, without masking, the assignment algorithm would have to choose between the two.

VI.2.2.4 Final cost matrix

Rather than choosing a single metric to construct the cost matrix, we merge positional and visual information to define a more robust hybrid cost.

Let $(\mathbf{C}^{\text{pos}}, \mathbf{C}^{\text{EMC2}}, \mathbf{C}^{\text{vis}}) = (\mathbf{C}^k)_{1 \leq k \leq K}$ be the different K candidate cost matrices. We first normalize each cost matrix in [0, 1]: $\hat{\mathbf{C}}^k = \frac{\mathbf{C}^k - \min(\mathbf{C}^k)}{\max(\mathbf{C}^k) - \min(\mathbf{C}^k)}$, before merging costs them via a weighted average:

$$\mathbf{C}_{ij}(\boldsymbol{\beta}) = \begin{cases} \infty & \text{if } M_{ij} = 1 \\ \sum_{k=1}^K \boldsymbol{\beta}_k \hat{\mathbf{C}}_{ij}^k & \text{otherwise} \end{cases} \tag{VI.4}$$

where $\boldsymbol{\beta} = [\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K]$ are the weights associated to each metric. These weights have to be tuned for each specific tracking problem, and we present in Chapter VI.2.3 an optimization algorithm to learn a nearly optimal mixture of costs from few annotated tracklets links.

VI.2.2.5 Tracklet Stitching Evaluation

We evaluate tracklet stitching performance by assessing the ability to recover ground-truth links from the set of predicted links. To this end, we manually annotated a video and its associated tracklets, resulting in a set of ground-truth links $\mathcal{L} = \{(i \to j) \in [\![1, N]\!]^2\}$. For a given cost matrix $\mathbf{C}$ and non-linking cost η, our tracklet stitching framework predicts a set of links $\hat{\mathcal{L}}(\mathbf{C}, \eta)$, which we compare to $\mathcal{L}$.

We report classical retrieval metrics: precision, recall, and F1-score, defined as

$$\mathrm{P}(\mathbf{C}, \eta) = \frac{\left|\hat{\mathcal{L}}(\mathbf{C}, \eta) \cap \mathcal{L}\right|}{\left|\hat{\mathcal{L}}(\mathbf{C}, \eta)\right|}, \tag{VI.5}$$

$$\mathrm{R}(\mathbf{C}, \eta) = \frac{\left|\hat{\mathcal{L}}(\mathbf{C}, \eta) \cap \mathcal{L}\right|}{|\mathcal{L}|}, \tag{VI.6}$$

$$\mathrm{F1}(\mathbf{C}, \eta) = 2 \cdot \frac{\mathrm{P}(\mathbf{C}, \eta)\,\mathrm{R}(\mathbf{C}, \eta)}{\mathrm{P}(\mathbf{C}, \eta) + \mathrm{R}(\mathbf{C}, \eta)}. \tag{VI.7}$$

The choice of $\eta \in [0, 1]$ plays a critical role. Small values of η favor high-precision but low-recall predictions (fewer links are accepted), while larger values yield higher recall at the expense of precision.

To obtain a performance measure independent of η, we compute the Average Precision (Links-AP), corresponding to the area under the precision–recall curve, as η varies from 0 to 1:

$$\text{Links-AP}(\mathbf{C}) = \sum_{\eta_p=0}^{1} \mathrm{P}(\mathbf{C}, \eta_p)\left(\mathrm{R}(\mathbf{C}, \eta_p) - \mathrm{R}(\mathbf{C}, \eta_{p-1})\right). \tag{VI.8}$$

Finally, for our method, the best cost-mixture weights $\boldsymbol{\beta}^\star$ are selected by grid search, optimizing the Links-AP metric. An alternative, annotation-efficient learning strategy is presented in the following Section.

VI.2.3 Costs mixture weights learning

In the previous Section, we find (nearly) optimal weights $\boldsymbol{\beta}^\star$ by grid-search which require a complete annotation of tracklet links for at least hundreds of frames.

We present here an alternative to grid search using much fewer manually annotated links so that the weights $\boldsymbol{\beta}$ of each cost can be easily tuned on new data. Let's assume to have annotated only a small subset of the links $\tilde{\mathcal{L}} \subseteq \mathcal{L}$ of cardinal $|\tilde{\mathcal{L}}| \ll |\mathcal{L}|$. Our approach then focuses on a single linear optimization problem to find suitable weights $\boldsymbol{\beta}$, approaching the performance of the optimal ones $\boldsymbol{\beta}^\star$.

For each manually-annotated link $(i \to j) \in \tilde{\mathcal{L}}$, the cost $\mathbf{C}_{ij}(\boldsymbol{\beta})$ should be smaller than the cost of any concurrent link. Formally, we define for a given link $(i \to j)$ the set of concurrent links as:

$$\overline{\mathcal{L}}(i \to j) = \{(p \to q) |\, p = i \text{ xor } q = j\}, \tag{VI.9}$$

in other words, all links that share a common tracklet with $(i \to j)$.

An ideal cost matrix should verify that for all annotated links in $\tilde{\mathcal{L}}$, the associated cost should be minimal, compared to the other concurrent links, that corresponds to the following minimax optimization problem:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^K} \max_{\substack{(i \to j) \in \tilde{\mathcal{L}} \\ (p \to q) \in \overline{\mathcal{L}}(i \to j)}} \mathbf{C}_{ij}(\boldsymbol{\beta}) - \mathbf{C}_{pq}(\boldsymbol{\beta}) \tag{VI.10}$$

By introducing a margin ϵ, we solve the equivalent linear minimization problem with a Simplex:

$$\begin{array}{ll} \min_{\epsilon\in\mathbb{R},\boldsymbol{\beta}\in\mathbb{R}^K} & \epsilon \\ \text{with} & \boldsymbol{\beta}^\mathsf{T}\mathbf{1}_K = 1 \text{ and } \boldsymbol{\beta} \geq 0 \\ & \forall (i \to j) \in \tilde{\mathcal{L}},\ \forall (p \to q) \in \overline{\mathcal{L}}(i \to j), \\ & \mathbf{C}_{ij}(\boldsymbol{\beta}) \leq \mathbf{C}_{pq}(\boldsymbol{\beta}) + \epsilon \end{array} \tag{VI.11}$$

VI.2.3.1 Weights Learning Evaluation

To evaluate our learning of the cost mixture weights $\boldsymbol{\beta}$, we rely on our annotated ground truth links $\mathcal{L}$. We randomly sample a subset $\tilde{\mathcal{L}}$ of $|\tilde{\mathcal{L}}|$ links and learn the weights $\boldsymbol{\beta}$ through our linear optimization problem (Eq. VI.11). Then, we evaluate the resulting cost $\mathbf{C}(\boldsymbol{\beta})$ on all the links, measuring the Links Average Precision on the link retrieval task. The results is compared with the AP measured with the optimal weights $\boldsymbol{\beta}^\star$. We repeat this operation 50 times for each given number of annotated links $|\tilde{\mathcal{L}}| < |\mathcal{L}|$ to account for the variability of the method due to link sampling.

VI.2.4 Learning visual features

Visual comparison between target could increase the accuracy and robustness of tracklet stitching. But this requires defining a visual embedding of the tracked objects. In biological tracking, several method have used *ad hoc* features (usually the shape, size or intensity of the target) [27,123,186,187]. Instead, we propose to use self-supervised learning to automatically learn the visual features that are relevant to this tracklet-stitching task.

We train a CNN encoder φ to visually discriminate target using only tracklet knowledge and thus requiring no human annotations. For each tracklet position $\mathbf{w}_t^i$, let P_t^i be the $L \times L$ spatial image patch extracted from frame t and centered on $\mathbf{w}_t^i$ (Figure VI.1.c shows different patches around target for three tracklets). Each patch is encoded by the CNN into discriminative features $\mathbf{f}_t^i = \varphi(P_t^i) \in \mathbb{R}^p$. The training goal is to produce similar features for patches from the same tracked target, but distant features for different targets.

Training Training is done on a single video with its associated tracklets. We extract the patches corresponding to all the defined positions of all the tracklets. Tracklets are divided into training (70%) and validation (30%) tracklets based on their position. We ensure that the training patches do not overlap with any of the validation ones.

The CNN is trained with the triplet loss [188] which has been introduced for learning similarities and dissimilarities between images. Let $\{P_a, P_p, P_n\}$ be a triplet of patches (anchor, positive, negative), and $\{\mathbf{f}_a, \mathbf{f}_n, \mathbf{f}_p\}$ their respective features. Then the triplet loss with margin m is defined as:

$$L\left(\mathbf{f}_a, \mathbf{f}_p, \mathbf{f}_n\right) = \max\left(||\mathbf{f}_a - \mathbf{f}_p|| - ||\mathbf{f}_a - \mathbf{f}_n|| + m, 0\right) \tag{VI.12}$$

Intuitively, the loss increases when the anchor features are closer to the negative ones than the positive ones. We therefore generate triplets so that the positive and anchor patches are from the same target, which is different than the negative one: we first sample uniformly an anchor tracklet γ^i and two random patches P_a and P_p from it (different frames of the same tracklet). Then, we sample a tracklet γ^j which overlaps temporally with γ^i. This prevents γ^i and γ^j from tracking the same target. Finally, we sample a random patch P_n from γ^j (see Figure VI.1.d).

Validation We define a pretext and unsupervised task to validate the training on our validation tracklets. The global visual aspect of a tracklet is characterized by the average of its patch features $\bar{\mathbf{f}}^i = \frac{1}{t^i_{\mathrm{d}} - t^i_{\mathrm{b}} + 1} \sum_{t=t^i_{\mathrm{b}}}^{t^i_{\mathrm{d}}} \mathbf{f}^i_t$.

For each tracklet γ^i, the encoder φ should cluster all the tracklet features $F^i = \{\mathbf{f}^i_t\}_{t \in [\![t^i_{\mathrm{b}}, t^i_{\mathrm{d}}]\!]}$ around the average features $\bar{\mathbf{f}}^i$. To evaluate this, we retrieve the n-closest features $\hat{F}^i(n) \subset \bigcup_{i'} F^{i'}$ among all computed features in the validation set. If the encoder is well trained, the tracklet features should be retrieved before the features of the other tracklets. As n increases, the recall increases and the precision reduces. We therefore define our validation metric as the mean Average Precision (mAP) of this retrieval task. Formally, it writes:

$$\mathrm{mAP} = \frac{1}{N} \sum_{i=1}^{N} \sum_{n=1}^{\infty} \mathrm{P}\left(\hat{F}^i(n), F^i\right) \cdot \left[\mathrm{R}\left(\hat{F}^i(n), F^i\right) - \mathrm{R}\left(\hat{F}^i(n-1), F^i\right)\right] \tag{VI.13}$$

Implementation details We assume a homogeneous target spatial distribution and set the patch size L to be ~ 3 times the average closest target distance ($L = 32$ pixels in our case) to include neighboring targets in each patch. This allows the CNN model to encode the target-specific shape, noise and texture, as well as the presence of neighboring targets. Patch intensities are normalized using the overall pixel mean and standard deviation in the training dataset. We use a default ResNet18 [115] architecture. After the last convolutional layer, adaptive average pooling is used, and a final linear layer projects the features onto a $p = 50$-dimensional space.

A batch size of 512 is used, and learning rate is tuned on the validation set from $[10^{-5}, 10^{-3}]$. We used a fixed version of Adam [189] with weight decay (10^{-2}): AdamW [190]. The momenta are set to 0.9 and 0.999 and the margin of the triplet loss is set to $m = 1$.

The CNN is trained for 20,000 steps with validation every 100 steps, the best network on validation set is selected. As patches are small, a full training lasts less than 2h on a single A100 Nvidia GPU.

VI.3 Results

We validate our method on time-lapse fluorescence sequences of *Hydra Vulgaris* neurons (Table VI.1) acquired with a spinning disk confocal microscope, using two fluorophores: GCaMP for calcium imaging (intermittent detectability) and TdTomato for nuclear labeling (stable).

Across the four videos, tracklets were generated with the Wavelet spot detector [24] followed by EMHT linking algorithm [26]. For Video 1, we manually annotated all 784 tracklets, yielding a total of $|\mathcal{L}| = 353$ ground-truth links, which served as the basis for evaluating the different stitching methods.

VI.3.1 Visual features learning

Table VI.1 reports the clustering performance of our CNN on tracklet patches. An individual CNN was trained independently for each corresponding video.

As expected, the task is inherently easier when the number of tracklets is small and tracklets are short, making direct performance comparisons across videos irrelevant. Still, a consistent trend can be observed: the CNN produces more robust and discriminative embeddings than simply using raw pixel values (no encoder). While raw features perform reasonably well given

Table VI.1: Data and self-supervised visual training results on *Hydra Vulgaris*. Hydra is imaged using two different fluorophores: GCaMP or TdTomato. We compare the mean Average Precision (mAP) of the features retrieval validation task with our trained contrastive features (ResNet18) versus *raw* features (flatten images as features).

Video	**1**	**2**	**3**	**4**
Nb. Frames	251	401	401	1000
FOV size	1024×1024	669×488	669×488	1024×898
Fluorophore	GCaMP	GCaMP	TdTomato	TdTomato
Nb. tracklets	784	1275	1763	4022
Median tracklet length	27	19	39	68.5
Nb. patches	45k	75k	162k	768k
mAP (Raw)	98.2%	96.6%	91.5%	77.9%
mAP (ResNet18)	**99.3%**	**98.4%**	**94.8%**	**97.6%**

Table VI.2: Tracklet links retrieval: We report the Average precision (Links-AP) of the different cost matrix methods on Video 1, which was fully annotated with 353 ground-truth links. We also report the impact of temporal masking in our LAP framework.

Cost	**Masking**	**Links-AP**
Position-only: $\mathbf{C}^{\text{pos}}$ [27]	Yes | No	49.1% | 22.6%
Motion guided: $\mathbf{C}^{\text{EMC2}}$ [28]	Yes | No	84.7% | 41.2%
Visual-only: $\mathbf{C}^{\text{vis}}$ (Ours)	Yes | No	64.6% | 26.8%
Best Mixture: $\mathbf{C}(\boldsymbol{\beta}^\star)$ (Ours)	Yes	**92.0%**

the simplicity of the patches, CNN-based features significantly outperform them for both fluorophores (GCaMP and TdTomato). Importantly, in the most challenging case (Video 4), raw features suffer a sharp drop in performance, whereas the learned features remain stable, indicating the increased robustness of the CNN representations.

VI.3.2 Tracklet stitching

Results for tracklet stitching on Video 1 are reported in Table VI.2 and Figure VI.2.a. We first observe that temporal masking substantially improves performance: for example, the average precision of the EMC2 cost nearly doubles when temporal masking is applied.

Regarding the different cost functions, our CNN-based visual distance outperforms the naive positional distance [27], improving the Links-AP from 49.1% to 64.6%. However, it remains below EMC2 [28], which achieves 84.7% AP. This behavior is expected in the context of Hydra neuron tracking: visual appearance alone provides limited discriminative power, as some neurons share similar visual aspects, while being spatially distant. In contrast, positional and motion cues remain highly informative.

Combining both sources of information proves most effective. Our best cost mixture $\mathbf{C}(\boldsymbol{\beta}^\star)$ reduces the linking errors of EMC2 by a factor of almost two, achieving 92% AP (8% remaining

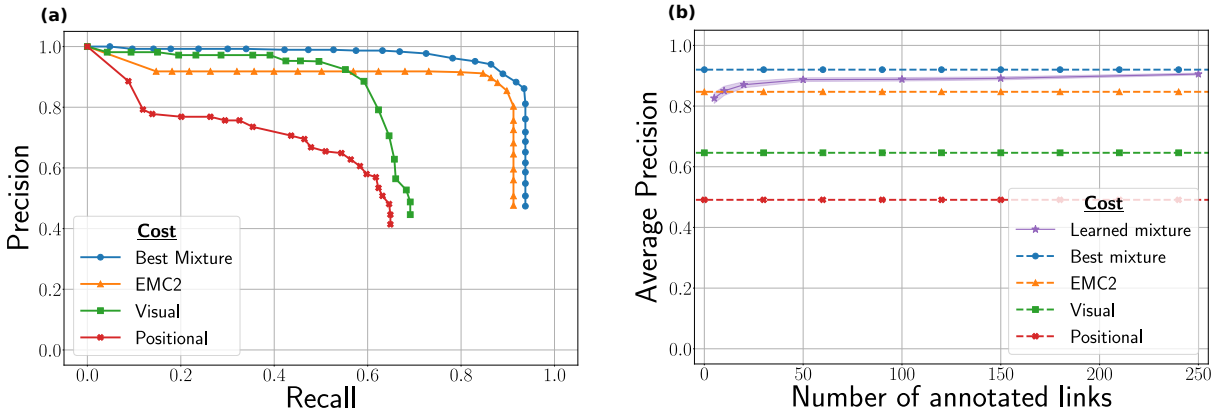


Figure VI.2: Tracklet stitching results on Video 1. **(a)** Precision-Recall curves as η increases from 0 to 1 for different cost matrix methods. Our best mixture found by grid-search ($\boldsymbol{\beta}^\star$) outperforms all other approaches, for any value of η. **(b)** Average precision (Links-AP) versus number of annotated links used to learn the mixture weights $\boldsymbol{\beta}$ (95% confidence interval on 50 trials). The other costs method are displayed as horizontal lines, as they are independent from the number of annotated links.

errors versus 15.3% for ECM2). Setting the non-linking threshold $\eta^\star$ to maximize the F1 score yields $\eta^\star = 0.05$, with a final performance of 90.1% F1 (94.1% precision and 86.4% recall).

VI.3.3 Grid search versus Weights learning

Figure VI.2.b shows the evolution of the Links-AP on Video 1, which contains 353 annotated ground-truth links, as a function of the number of annotated links $|\tilde{\mathcal{L}}| \in \{5, 10, 20, 50, 100, 250\}$ used to learn the weights $\boldsymbol{\beta}$ (Eq. VI.11). We compare the cost induces by the learned weights $\mathbf{C}(\boldsymbol{\beta})$ with the optimal one obtained by grid search on all the annotations $\mathbf{C}(\boldsymbol{\beta}^\star)$, as well as with the individual cost methods.

Remarkably, with only 10 annotated links, our algorithm learns weights $\boldsymbol{\beta}$ that already outperform any single cost metric. With more annotations, performance steadily improves and rapidly approaches the optimum found by grid search.

Grid search yields the best weights $(\beta^\star_{\text{EMC2}}, \beta^\star_{\text{vis}}, \beta^\star_{\text{pos}}) = (0.9, 0.1, 0.0)$. This indicates that, on Hydra data, motion-corrected positions from EMC2 provide the dominant information for tracklet stitching, while visual features contribute to resolving ambiguities between spatially close neurons. In contrast, the raw positional cost is not used, reflecting the strong deformations of the animal. When learned from annotations, the weights evolve from $(0.59, 0.22, 0.19)$ with only 5 links, to $(0.78, 0.14, 0.08)$ with 20 links, and finally converge toward $(0.87, 0.11, 0.02)$, very close to the optimal $\boldsymbol{\beta}^\star$.

Since the optimal weights $\boldsymbol{\beta}$ are dataset-dependent, these results highlight the strength of our learning approach: it can reliably estimate near-optimal mixtures from only a handful of annotated links, making it practical for adapting the framework to new datasets with minimal supervision.

VI.4 Discussion

In this Chapter, we proposed a post-processing method to robustly stitch tracklets of intermittent targets in Multiple Object Tracking. Our main contribution is the integration of both visual

and positional features into the tracklet-to-tracklet cost, extending the EMC2 framework [28], which uses a Linear Assignment Problem (LAP) to optimally merge fragmented tracklets into full trajectories.

For the visual component, square patches centered on detections are extracted and embedded into discriminative features using a ResNet [115] trained with contrastive self-supervised learning. For the positional component, we re-use previously defined motion costs, from raw Euclidean distance [27] to the motion corrected EMC2 [28]. To combine both sources of information, we introduce an optimization algorithm that learns the optimal linear mixture of visual and positional costs from only a few manually annotated links. On fluorescence recordings of *Hydra Vulgaris* neurons, our framework reduces stitching errors by a factor of two compared to EMC2.

In constructing the visual metric, we relied on a standard ResNet architecture originally designed for natural images. However, in many biological contexts, visual features are less complex than in natural scenes. We therefore expect that smaller architectures could provide comparable accuracy with reduced computational cost. In addition, our training relied on the triplet loss [188]; more recent contrastive objectives such as InfoNCE [191], NT-Xent [192], or Barlow Twins [193] have demonstrated superior performance and may further improve the discriminative power of learned features. Exploring these alternatives remains an interesting perspective for future work.

This preliminary study did not include comparisons with handcrafted visual features [186], which could potentially yield similar performance. Nevertheless, self-supervised learning offers the advantage of generalization across datasets without requiring dataset-specific expert knowledge.

Finally, the proposed optimization strategy for learning cost combinations from very few annotations is particularly promising. By enabling near-optimal parameter tuning with minimal manual input, it increases the versatility of the framework and facilitates its application to new biological datasets with limited supervision.

Chapter VII

Conclusion and Perspectives

VII.1 Applications

VII.1.1 Hydra Vulgaris

At the beginning of this manuscript (see Chapter II.4), we showed that standard MOT approaches struggle to track Hydra neurons reliably. Large and sudden contractions of the animal, combined with missed detections and false positives, introduce frequent tracking errors. Figure II.8 highlighted a typically hard example: the contraction of a Hydra tentacle. Figure VII.1 illustrates how our newly developed algorithms, KOFT and KOFT-MHT, achieve much greater robustness on the same hard example, enabling the correct tracking of most neurons despite the contraction event. To facilitate visualization, we uploaded videos[1] of the first 10s of this tentacle example tracked with the different tracking methods.

While KOFT resolves most motion-related errors, in this extreme case optical flow slightly underestimates the downward displacement, leading to several identity switches. For instance, tracks 1 and 2 swap identities with tracks 3 and 4. These switches occur because the underestimation of motion coincides with missed detections. Nevertheless, thanks to the optical-flow-enhanced motion model, KOFT terminates the affected tracks instead of reassigning them incorrectly. In contrast, eMHT and TrackMate mistakenly link these tracks to detections above, triggering cascades of failures across other non-displayed tracks.

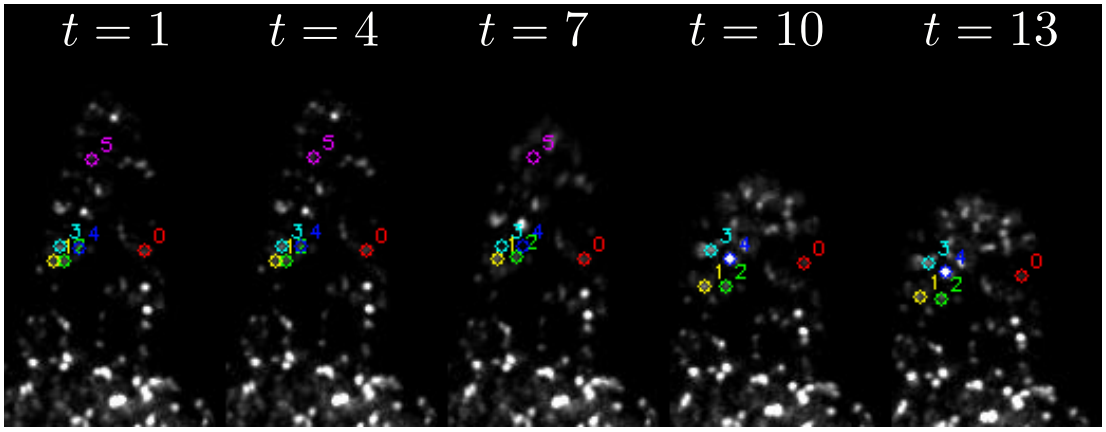
By integrating our Bayesian and Multiple Hypothesis Tracking framework, KOFT-MHT becomes more robust to detection errors. In particular, it leverages subsequent frames to identify that track 2 was only temporarily missed, successfully recovering the trajectories of neurons 2 and 4 without identity switches. For neuron 3, which truly goes out of focus, KOFT-MHT still terminates track 1 instead of track 3, a limitation caused by the optical flow underestimation. Moreover, KOFT-MHT can bridge longer gaps than KOFT; for example, neuron 0 is terminated too early by KOFT, whereas KOFT-MHT maintains the track correctly.

We further evaluated KOFT-MHT against eMHT on a 100-second TdTomato video (1000 frames at 10 Hz). To bridge short gaps caused by neurons going out of focus, we applied the eMC2 [28] stitching procedure, limited to gaps shorter than 30 frames. Following the practice of TRASE-IN [22], only tracks spanning more than 800 frames were retained, as full-length raw calcium traces are required for downstream analysis.

Under these conditions, KOFT-MHT successfully reconstructed 446 long tracks, compared

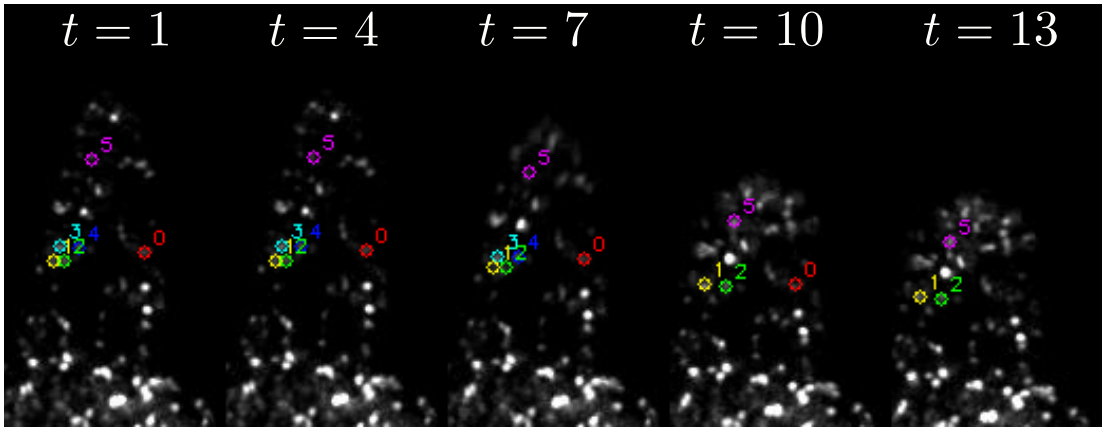
[1] `https://partage.imt.fr/index.php/s/7cYJXqB2Qwao3a6`

(a)



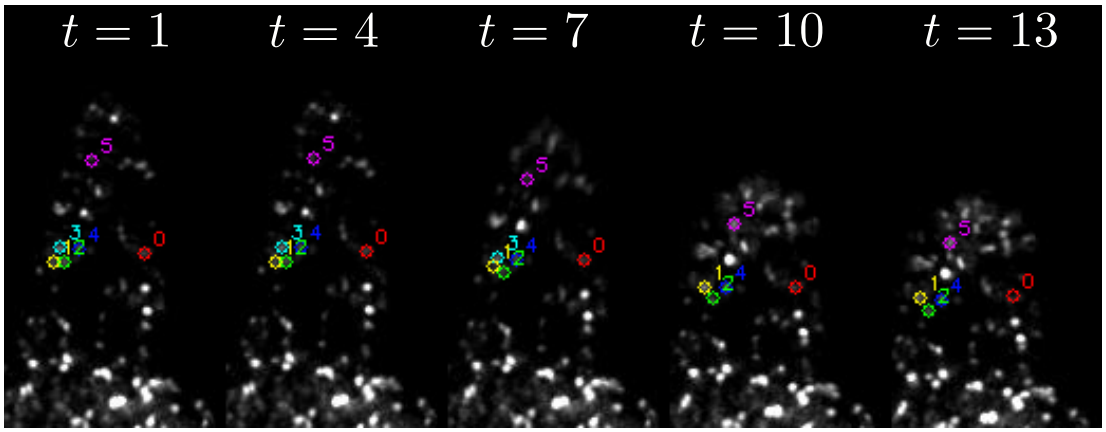
o[5] eMHT track (id)

(b)



o[5] KOFT track (id)

(c)



o[5] KOFT-MHT track (id)

Figure VII.1: Remaining tracking errors of our newly developed MOT approaches on the contracting tentacle example of Figure II.8.
(a) Six representative tracks obtained with EMHT [26], reproduced from Figure II.8 for comparison.
(b) The same six tracks with KOFT. Here, track 5 is perfectly reconstructed. Track 1 (resp. 2) still swaps with track 3 (resp. 4) at $t = 7$, but unlike in EMHT, tracks 3 and 4 are terminated rather than being wrongly reassigned. Track 0 is accurately tracked up to frame 10, after which consecutive missed detections lead to premature termination before frame 13.
(c) The same six tracks with KOFT-MHT. In this case, tracks 0, 2, 4, and 5 are reconstructed without error. Track 1 still briefly swaps with track 3, which then terminates. Thanks to the Bayesian solver combined with the optical-flow-enhanced motion model, most errors induced by synchronous missed detections during tentacle contraction are corrected.

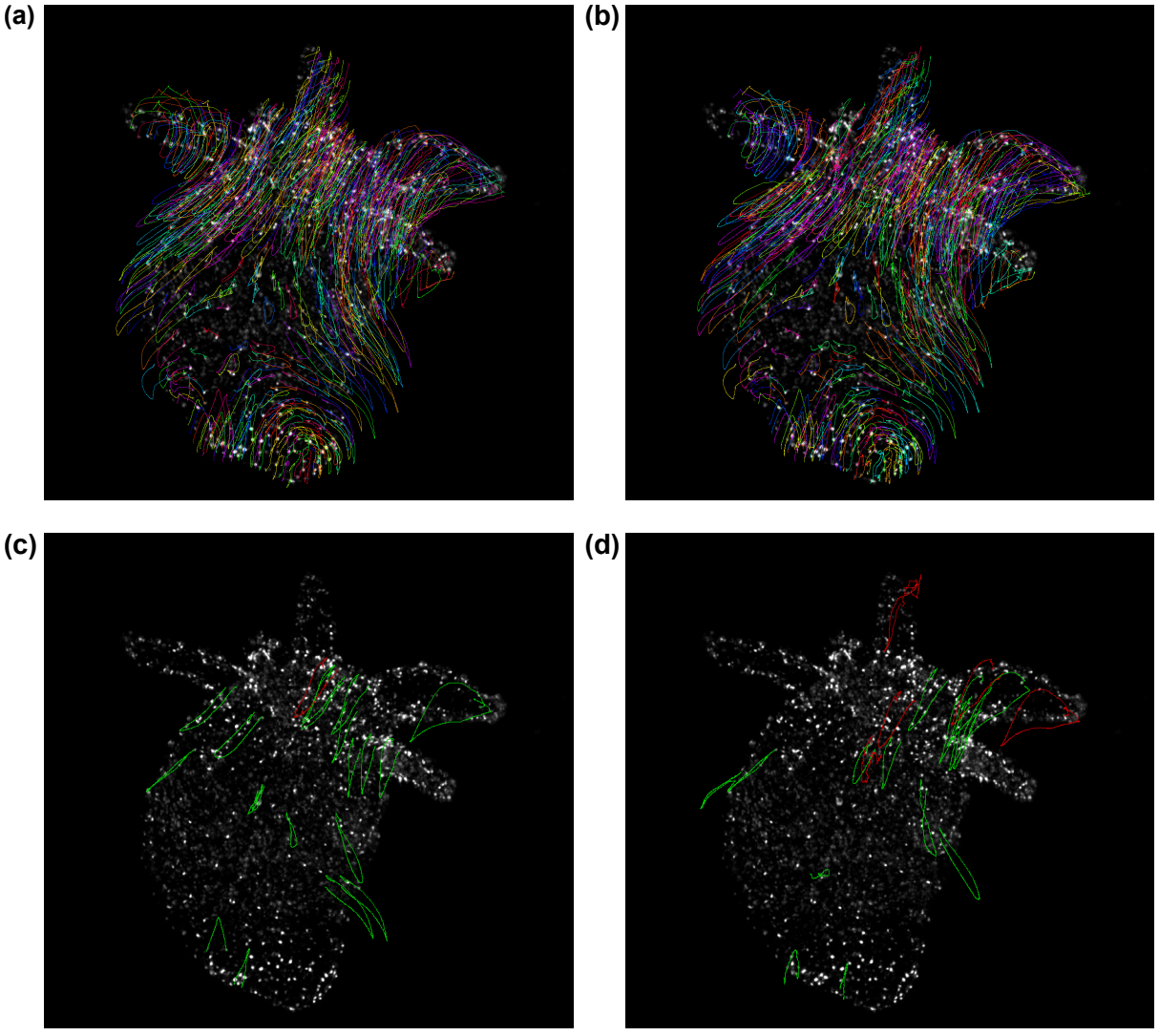


Figure VII.2: Long-term tracking of TdTomato neurons over 1000 frames (100s).
(a) KOFT-MHT recovers 446 long tracks (that spans over more than 800 frames), overlaid on the first frame of the sequence. **(b)** EMHT produces 421 such tracks under the same conditions. **(c)** Among 20 randomly sampled KOFT-MHT tracks, only one (in red) is erroneous, while 19 are correct (green). **(d)** In contrast, EMHT produces 5 erroneous tracks out of 20.

with 421 for EMHT, representing an improvement of 25 trajectories. In both cases, this corresponds to approximately half of the ∼800 visible neurons. Qualitative inspection (Figure VII.2) shows that KOFT-MHT not only produces slightly more complete trajectories, in particular in the tentacles of the Hydra, but also yields substantially fewer errors: among 20 randomly sampled tracks, only one was incorrect with KOFT-MHT, versus five with EMHT. These results highlight that KOFT-MHT greatly facilitates downstream neuronal activity analysis by providing both more complete and more reliable long-term trajectories.

Overall, our proposed methods significantly outperform existing MOT approaches for Hydra neuron tracking. These algorithms are implemented in the Python library `ByoTrack` and are already in use in our laboratory and in Rafael Yuste's lab (Columbia University) for neuronal activity studies, as a replacement of the previously used EMHT algorithm [22].

VII.1.2 Cell Tracking Challenge

Although this thesis focuses on Hydra neuron tracking, we also aimed to design general-purpose tracking methods applicable to other biological targets. After developing KOFT, we participated in the Cell Linking Benchmark[2] (CLB) of the Cell Tracking Challenge (CTC) [59] to evaluate KOFT performance on datasets involving cell division. In this section, we describe our minor adaptation of KOFT to support mitosis and to automatically tune parameters across a wide range of cell tracking scenarios.

VII.1.2.1 Cell Linking Benchmark

In the CLB, the linking part of tracking-by-detection algorithms (see Chapter II) are evaluated on 16 datasets (13 experimental and 3 synthetic, in 2D and 3D), each providing approximate cell segmentation. Each dataset contains 4 videos: 2 with public tracking annotations (typically used for training deep models) and 2 with private annotations reserved for evaluation. Across datasets, cells vary greatly in size, motion, and visual appearance (see Figure I.14), and mitosis occurs only in a subset of datasets. A successful algorithm must therefore handle both consistent tracking and division events.

Performance is evaluated with the LNK metric, which measures the normalized AOGM [194], i.e. the minimal number of operations required to transform the predicted tracking graph into the ground truth one. In addition, four biologically meaningful measures are aggregated into the BIO score, which reflects the ability to reconstruct full trajectories and capture mitoses. The final score (OP) is the average of LNK and BIO.

VII.1.2.2 KOFT Adaptation

We adapted KOFT (Chapter IV) to the CTC datasets with three minor modifications:

1. Brownian motion. Cells in the CTC datasets typically exhibit Brownian-like dynamics rather than directed motion. We therefore used the Brownian Kalman filter model described in Chapter IV.2.3.

2. Distance-based association. We replaced our likelihood-based association (maximizing measurement likelihood) with a simpler Euclidean distance minimization criterion. For each active track $\gamma^i_{1:t}$, with its predicted state $\bar{\mathbf{x}}^i_{t+1}$ and detection $\mathbf{z}^j_{t+1}$, we compute the linking cost

[2] `https://celltrackingchallenge.net/latest-clb-results/`

as:

$$\mathbf{C}_{ij} = \left\| \mathbf{H}^{\text{pos}} \bar{\mathbf{x}}_{t+1}^{i} - \mathbf{z}_{t+1}^{j} \right\|_2 . \tag{VII.1}$$

This choice simplifies the estimation of the association threshold η, which is now expressed in pixels.

3. Mitosis detection. To detect splitting events, we introduced a second linking step between already linked tracks and unmatched detections. We biased associations toward plausible mitoses by incorporating size information. If track i (size ρ_i) is linked to detection j (size ρ_j), the cost of linking i to another detection k (size ρ_k) is:

$$\bar{C}_{ik} = C_{ik} \cdot \gamma_{jk} \cdot \gamma'_{ijk},$$

where $\gamma_{jk} = \frac{\max(\rho_j, \rho_k)}{\min(\rho_j, \rho_k)}$ penalizes size disparity between the two children, and $\gamma'_{ijk} = \frac{\max(\rho_i, \rho_j + \rho_k)}{\min(\rho_i, \rho_j + \rho_k)}$ penalizes size inconsistencies between the parent and children. This step is solved with the Jonker–Volgenant algorithm [30].

VII.1.2.3 Parameter Estimation

To ensure generality, we designed simple rules to automatically set KOFT parameters from detection statistics. Specifically, we compute: **(1)** the average cell radius ρ, **(2)** the average nearest-neighbor distance d_{closest}, and **(3)** the ratio $\frac{M_T}{M_1}$ between the final and initial number of detected cells.

Parameters are then set as follows:

- Kalman uncertainties: $\sigma_{\text{pos}} = \frac{\rho}{2}$, $\sigma_{\text{vel}} = \sigma_{\text{acc}} = 3\rho$ (low reliance on optical flow, since Farneback performs poorly on these datasets).
- Farneback window size: $w = \max(10, \rho/2)$.
- Association threshold: $\eta = \max(3\rho, d_{\text{closest}})$.
- Splitting events considered only if $\frac{M_T}{M_1} > 30\%$.
- $\tau_{\min} = 1$ (no false positives in CLB), $\tau_{\max} = 1$ (few missed detections in CLB).

We further refined some parameters ($\tau_{\max}$, η or split handling) with a grid search on the two training videos of each dataset. As OpenCV's Farneback does not support 3D, we used SKT in 3D while keeping the rest of the framework unchanged. We emphasize that our KOFT implementation does also support 3D optical flow methods, but we did not manage to find a suitable optical flow in 3D in the restricted time that we had for the submission.

VII.1.2.4 Results

As shown in Table VII.1, KOFT ranked first in the general submissions, achieving the highest OP score across the 13 experimental datasets. Remarkably, it also outperformed supervised methods such as TrackAstra, zTrack, and RWTH-GE. This is likely due to the limited training data (two videos per dataset) and occasional annotation errors, which may have hindered supervised methods. Notably, KOFT achieved an 86.2% BIO score despite its simple mitosis module, while all competing methods remained below 80%.

Table VII.1: Cell Linking Benchmark results (general submissions). LNK, BIO, and OP scores are averaged over the 13 experimental datasets.

Method	**LNK**	**BIO**	**OP_{CLB}**
TrackMate [128] (Baseline)	96.4%	74.7%	85.5%
TrackAstra [64] (EPFL-CH)	97.7%	79.4%	88.6%
Baxter [119] (KTH-SE)	95.8%	75.2%	85.5%
TrackTour [105] (MON-AU)	91.4%	67.4%	79.4%
zTrack (SIAT-CN)	96.6%	71.5%	84.0%
RWTH-GE	97.2%	69.9%	83.5%
KOFT [66] (PAST-FR)	**98.4%**	**86.2%**	**92.3%**

Across all 16 datasets, KOFT ranked first or second on the LNK metric in 13 cases, while TrackAstra (the next-best method) achieved second place or better in only 8 cases. This outlines the robustness of our approach on such cell tracking datasets.

We anticipate that our Bayesian tracker KOFT-MHT would achieve even higher scores (though it was not developed at the time of submission). However, improvements may be modest, since CTC segmentations are highly accurate (with not a single false positives and very few misses), and extending MHT to explicitly handle cell division [132, 134] is substantially more complex than our minor modification to KOFT.

VII.1.3 Broader Biological Applications

Since the development of our SKT and KOFT tracking algorithms, we have distributed them to other biological researcher for very different applications. Figure VII.3 shows the variety of the targets in our collaborations, and below is the list of (known) applications of our tracking algorithms:

- Hydra neurons (J.-C. Olivo-Marin lab, Institut Pasteur & Yuste lab, Columbia University)
- Zebrafish neurons (L. Hernandez-Nunez lab, Harvard University & Naumann Lab, Duke University)
- *Entamoeba histolytica* parasites (J.-C. Olivo-Marin lab, Institut Pasteur)
- *Toxoplasma gondii* parasites (I. Tardieux lab, IAB Grenoble)
- CCR5 membrane receptors (Jost Enninga lab, Institut Pasteur)
- Cell Tracking Challenge (see previous Section)
- T-cells (Exthymics, Californie)

This further indicates that our algorithms are indeed able to generalize to many different biological targets, broadening the impact of this thesis beyond our *Hydra Vulgaris* animal model.

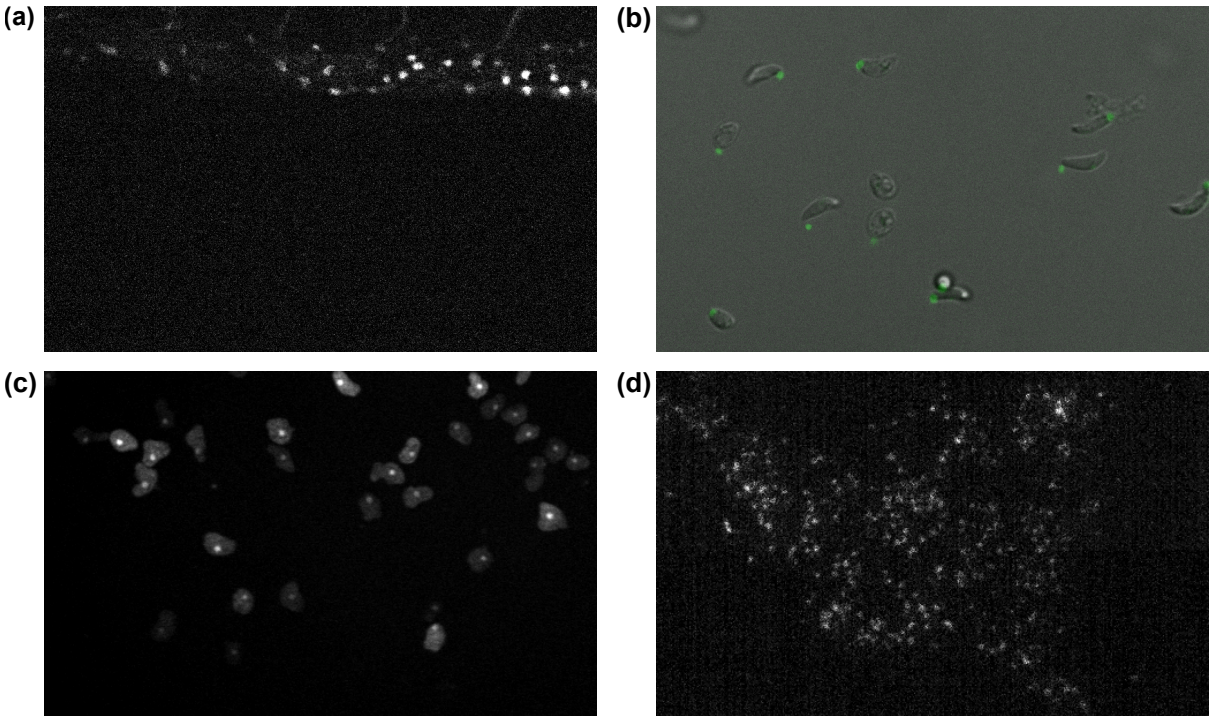


Figure VII.3: Examples of broader biological applications. **(a)** Zebrafish neurons, GCaMP 3D imaging with a Spinning Disk Confocal Microscope (from S. Zhang, L. Hernandez-Nunez lab, Harvard University) **(b)** *Toxoplasma gondii* parasites, Spinning disk confocal microscope (from G. Drumont, I. Tardieux lab, IAB Grenoble) **(c)** *Entamoeba histolytica* parasites, Confocal Microscope [195] (M. Manich, J.-C. Olivo-Marin lab, Institut Pasteur) **(d)** CCR5 membrane receptors, Fluorescence TIRF microscopy [196] (from M. Bilay and A. Brelot, Jost Enninga lab, Institut Pasteur)

VII.2 Thesis Summary

Deciphering neural coding in small behaving animals such as *Hydra vulgaris* begins with calcium fluorescence imaging of the entire animal. This produces time-lapse sequences in which neuronal fluorescence intensity correlates with activity. The first step in analyzing such data is to track all neurons across the sequence and extract their activity profiles. To ensure reliable downstream analyses, the tracking pipeline must be robust: it should maintain consistent neuron identities over long time spans to capture activity correlations, while avoiding identity switches that contaminate subsequent analyses.

In this thesis, we developed several methodological contributions to improve neuron tracking in *Hydra vulgaris*. These contributions can be grouped into four main axes:

(1) SINETRA synthetic datasets. To address the lack of annotated experimental datasets and of synthetic datasets resembling Hydra recordings, we developed *SINETRA*, a versatile simulator for multiple-object tracking. SINETRA generates synthetic targets embedded in deformable tissues, with realistic motion patterns obtained either from optical flow extracted from experimental Hydra videos or from elastic spring models driven by random contraction and elongation forces. The simulator produces 2D and 3D fluorescence videos corrupted with Poisson shot noise and additive background autofluorescence mimicking the synthetic animal body. Each video comes with full ground-truth annotations, enabling rigorous benchmarking of tracking algorithms under conditions that closely represent the challenges of Hydra imaging.

SINETRA thus provides a valuable foundation for developing and evaluating robust neuron tracking methods.

(2) MOT with optical flow-enhanced motion models. We observed that a key limitation of Bayesian tracking-by-detection methods lies in their motion models (usually Kalman filters), which predict each target's motion independently from past measured positions. These models are unable to anticipate sudden accelerations, that typically occur when the Hydra contracts. To overcome this, we introduced optical flow as an additional "sensor" in the Kalman filter framework, providing velocity estimates that anticipate future displacements. This optical flow-enhanced Kalman filter yields motion predictions that balance noisy velocity estimates with past positional observations. We integrated this model into an online frame-by-frame tracker, KOFT, that clearly outperformed similar and more advanced Bayesian algorithms on SINETRA and experimental Hydra data.

(3) Robust Bayesian Multiple Hypotheses Tracking: We developed a principled Bayesian tracking framework and an efficient Multiple Hypothesis Tracking (MHT) solution capable of robustly tracking numerous (>1000) biological targets. Building on the success of KOFT, we introduced its Bayesian MHT counterpart, KOFT-MHT, which combines our optical-flow–enhanced motion model with the Bayesian framework. This approach further improves robustness, particularly under noisy detection conditions, and consistently outperforms all previous methods on SINETRA. In parallel, our SKT-MHT tracker—which excludes optical flow (standard Kalman filtering)—outperformed state-of-the-art MHT frameworks, making it applicable in contexts where optical flow is not reliable, or computationally prohibitive, broadening the impact of our MHT method. Finally, our Bayesian trackers require only a few manual parameters, all of which can be estimated with minimal knowledge of the tracking algorithm, therefore extending their usability to non-expert users.

(4) Tracklet stitching with visual features. In calcium imaging with GCaMP fluorophores, neurons are only visible when active, leading to intermittent detectability and fragmented trajectories: MOT algorithms, including ours, often produce short tracklets corresponding to the periods of activity of the neuron. To reconstruct complete trajectories, we trained a convolutional neural network using self-supervised contrastive learning to discriminate fluorescent patches of neurons (32×32 pixels). We then solved the tracklet-to-tracklet association problem using the Jonker–Volgenant algorithm, combining learned visual embeddings with spatial and temporal cues. On experimental Hydra data, this approach reduced stitching errors by a factor of two when compared to previous methods.

All tools developed during this PhD are open source and integrated into the `ByoTrack` library, which interfaces with Icy, Fiji, and soon Napari. While specifically designed for *Hydra vulgaris*, our methods generalize to other systems, including *C. elegans* and Zebrafish. We deliberately focused on unsupervised or self-supervised approaches to ensure broad applicability across diverse biological targets, including cells, membrane receptors, parasites. Notably, our methods SKT/KOFT ranked first in the Cell Linking Benchmark of the Cell Tracking Challenge, outperforming both classical and supervised approaches.

Overall, this thesis contributes methodological innovations and practical tools that improve the reliability of biological video analysis. In particular, by enabling consistent tracking of neuronal activity, it contributes to a deeper understanding of neural dynamics in living animals.

VII.3 Limitations and Perspectives

While this thesis advances the field of biological multiple object tracking (MOT) through novel contributions, the scope of three and a half years inevitably left certain challenges unaddressed. Below, we outline the primary limitations of our work and propose key perspectives to further advance biological MOT, with a focus on neuron tracking.

Synthetic Simulator

Our synthetic simulator, SINETRA, currently models fluorescent images where neuronal activity dynamics are not modeled (TdTomato). A critical extension would be to design datasets with time-evolving individual neuronal activity, thereby increasing the realism to calcium imaging (GCaMP). This would increase the challenge of the tracking benchmark and enable the rigorous evaluation of tracklet stitching methods. For instance, our stitching method was validated on a single manually annotated video. Moreover it would allow for the integration of global neuronal activity models, such as those proposed by [23]. Such models could facilitate benchmarking of the entire analysis pipeline, from tracking to the clustering of neurons into functional ensembles.

Additionally, SINETRA could leverage recent advances in deep generative models to produce more realistic neuron shapes, moving beyond our current elliptic model. This enhancement would be particularly valuable for image-based trackers that rely on visual features for target re-identification across frames.

Optical Flow and Motion Modeling

Our work demonstrated that optical flow significantly enhances motion prediction and tracking performance, provided the flow estimates are sufficiently accurate. While the Farneback algorithm [179] proved robust for *Hydra* and SINETRA datasets, its performance degraded on other datasets (e.g., CTC), necessitating reduced confidence in our KOFT algorithm and limiting the optical flow-enhanced motion model accuracy. This limitation underscores the need to benchmark a broader range of optical flow methods, particularly in 3D, where computational complexity and the additional degree of freedom pose greater challenges.

An alternative approach on such datasets could involve integrating methods like EmbedTrack [63], which jointly learns cell segmentation and future displacement. Such methods could replace optical flow in our framework. We emphasize that KOFT and KOFT-MHT would directly benefit from any advances in optical flow-like algorithms.

Beyond the 1-to-1 Association Assumption

Our Bayesian trackers operate under the 1-to-1 association assumption, limiting their robustness when detections are merged or split—a common issue with unsupervised segmentation methods like the Wavelet spot detector [24]. While deep learning-based detectors (e.g., StarDist [25]) mitigate such errors, we believe that removing this assumption within a Bayesian Multiple Hypothesis Tracking (MHT) framework would improve tracking robustness at the cost of generating more hypotheses and an increased computational complexity. While graph-based [91, 105, 116, 117] or online [197] approaches have explored this direction, to the best of our knowledge, no MHT framework currently supports multiple track-to-detection assignments. Addressing this gap represents a promising avenue for future work.

Handling Target Splitting and Merging

Our MHT trackers do not yet support target splitting and merging (different issue than detection splitting/merging, see Chapter II), which are essential for handling cell mitosis. Incorporating these—by generating corresponding tracking hypotheses [132, 134]—would enable our methods (KOFT-MHT and SKT-MHT) to compete in challenges like the Cell Tracking Challenge.

Integration of Visual Features in tracking

We demonstrated that visual features improve neuron identification in our tracklet stitching post-processing step. Extending this idea to the tracking phase could further differentiate between closely located neurons when motion models lacks precision. This concept is well-established in natural image tracking (e.g., DeepSort, StrongSort), where visual features complement positional Kalman filters [198, 199]. While we initiated this development, time constraints prevented its integration into this thesis. Future work could explore embedding visual features into our MHT framework to refine associations and prune hypothesis trees. Calibrating our contrastive CNN to output probabilistic estimates would also facilitate its integration into our Bayesian framework.

Graph-Based Extensions

Our Bayesian framework could be extended into a graph-based method under the Markovian cost assumption. While this assumption may be restrictive for directed motion, it could be compensated by optical flow corrections and the advantages of optimizing over the full temporal sequence. Unlike most graph-based approaches in biology—which rarely account for missed detections or rely on ad-hoc costs—our principled Bayesian formulation models missed detections as edges from t to $t+\tau$ with appropriate costs. For example, extending TrackTour [105] with our Bayesian costs could enhance its performance. Moreover, such graph-based method could directly tackles the tracklet-stitching problem by solving large gaps (> 50 frames), as long as the hardware memory constraints supports it.

Improving Detection for Tracking

The performance of tracking-by-detection algorithms is fundamentally constrained by detection quality. While deep learning-based detectors like StarDist [25] initially appeared sufficient for our *Hydra* use case, residual errors—particularly in tentacle regions—highlight the need for improved detection. Two promising directions include:

- **Video-based detection:** Leveraging temporal information across multiple frames to inform detection decisions.
- **Track-guided detection:** Using tracking information from previous frames to bias detection in subsequent frames.

Both approaches would violate the Bayesian MOT assumption of independent positional errors across frames. However, the potential benefits for detection accuracy likely outweigh the drawbacks of an imperfect tracking model.

Conclusion

These perspectives outlined above aim to address the current limitations of our work and push the boundaries of biological MOT. By refining synthetic data generation, motion modeling,

association strategies, and detection methods, we can further enhance the robustness and applicability of tracking frameworks.

Appendix A

Appendix: Additional results

A.1 KOFT versus Warp-Then-Track

KOFT uses optical flow as an additional noisy measurement within a Bayesian motion model (Kalman filter). A more straightforward alternative is to use optical flow for image registration: detections are warped onto a reference frame and then linked, either with overlap-based or distance-based frame-by-frame method [137, 148, 200] or with graph-based optimization, as in LINEAJEA [62]. We implemented such a strategy, termed *Warp-Then-Track* (WTT), where optical flow is used to warp detections prior to applying a standard tracking algorithm (SKT or EMHT).

Instead of warping images, which is computationally expensive and prone to interpolation artifacts that may degrade the detection process, we rather warp detections $Z_{1:T}$. Each detection $\mathbf{z}_t^j$ at time t is mapped onto the last frame T via the cumulative optical flow $\mathbf{\Phi}_{t,T}$ (computed iteratively from flows $(\mathbf{\Phi}_{i,i+1})_{t \leq i < T}$):

$$\hat{\mathbf{z}}_t^j = \mathbf{z}_t^j + \mathbf{\Phi}_{t,T}(\mathbf{z}_t^j).$$

This yields warped detections $\hat{Z}_{1:T}$ on which any tracking algorithm can operate.

Since warping suppresses target dynamics, the resulting motion is well described by a Brow-

Table A.1: Comparison of *Warp-Then-Track* (WTT) approaches with *KOFT* on SINETRA. We report the HOTA score [180] (at 2 pixels tolerance). Motion is modeled with Brownian, Directed, or Accelerated Kalman filters. In EMHT, an IMM combines Brownian and Directed models. Detections are obtained with the Wavelet spot detector [24]. For WTT, Farneback optical flow [179] is used to warp detections before tracking. KOFT instead integrates the same optical flow directly into its Bayesian motion model, with an uncertainty of $\sigma_{\text{vel}} = 1.0$.

SINETRA Scenario	Springs 2D	Hydra Flow	FPS
WTT-EMHT [26] (IMM)	$91.0 \pm 1.1\%$	$86.5 \pm 2.2\%$	2
WTT-SKT (Brownian)	$89.6 \pm 1.3\%$	$84.6 \pm 2.1\%$	8
WTT-SKT (Directed)	$82.4 \pm 2.4\%$	$71.6 \pm 2.4\%$	8
KOFT (Directed)	$95.6 \pm 0.6\%$	$93.4 \pm 1.9\%$	10
KOFT (Accelerated)	$\mathbf{95.7} \pm 0.6\%$	$\mathbf{94.0} \pm 1.8\%$	10

nian model. Accordingly, we tested WTT-SKT with Brownian and Directed Kalman filters, and WTT-eMHT with its standard IMM. For KOFT, we considered its default directed filter and tested an accelerated Kalman filter, which explicitly models near-constant acceleration.

Table A.1 summarizes the results. First, WTT significantly improves the performance of classical trackers compared to their non-warped versions (Table IV.2). For example, SKT nearly halves its error rate: from 80.5% to 89.6% HOTA on Springs 2D, and from 72.8% to 84.6% on Hydra Flow. WTT-eMHT slightly outperforms WTT-SKT, reflecting the benefit of multi-frame assignments (MHT) and IMM motion modeling.

Second, KOFT consistently outperforms WTT, reaching 95.6% HOTA on Springs 2D and 93.4% on Hydra Flow. Its advantage comes from three potential factors:

1. **Uncertainty modeling:** Unlike WTT, KOFT does not fully trust optical flow, but treats it as a noisy measurement. This makes it robust to poor flows. For example, under challenging conditions (low visibility $\alpha = 0.02$, $\Delta = 50.0$, Fake@80% detector), WTT-SKT and WTT-eMHT drop to 47.5% and 51.2% HOTA and are outperformed by their non-warped counterparts that reach 64.3% and 74.4% HOTA. In contrast, KOFT maintains 75.4% HOTA, outperforming all methods.
2. **Improved state estimates:** By integrating optical flow within Kalman filtering, KOFT produces more accurate position estimates than standard models applied after warping.
3. **Field-of-view limitations:** WTT relies on a single reference frame. Targets outside this frame (entering or leaving the field of view) cannot be tracked accurately. KOFT, operating online, has no such limitation.

Finally, WTT-SKT with a directed filter underperforms, with HOTA scores below 85%. This is expected, as warping removes motion dynamics, leaving a Brownian-like motion. In contrast, KOFT preserves the natural dynamics and using an accelerated model even provides slight additional gains (+0.1–1% HOTA).

In conclusion, while WTT improves classical trackers, our KOFT algorithm achieves higher accuracy and robustness. Remarkably, it even outperforms WTT combined with advanced MHT solvers, despite its relative simplicity. In essence, our optical-flow–enhanced Kalman filter provides advantages over raw optical flow that are analogous to the benefits of standard Kalman filtering over using raw positional measurements.

A.2 MHT - Impact of the Motion Model

We observed that eMHT outperformed our standard Bayesian tracker SKT-MHT in low-quality detection settings (F1 score $\leq 70\%$). In this section, we demonstrate that this advantage is largely due to the Interacting Multiple Model (IMM) filter implemented in eMHT, which combines both Brownian and Directed motion models. With a Brownian motion model, SKT-MHT consistently outperforms eMHT in low-quality detection settings.

We compared SKT-MHT and KOFT-MHT with both Brownian and Directed Kalman filters, and eMHT with Brownian, Directed, and IMM filters, using our SINETRA simulator (Springs 2D and Hydra Flow scenarios). Results are reported in Figure A.1.

The results confirm that Directed Kalman filters generally outperform Brownian ones in our synthetic scenarios under high-quality detections (F1 $\geq 90\%$). However, in the presence of

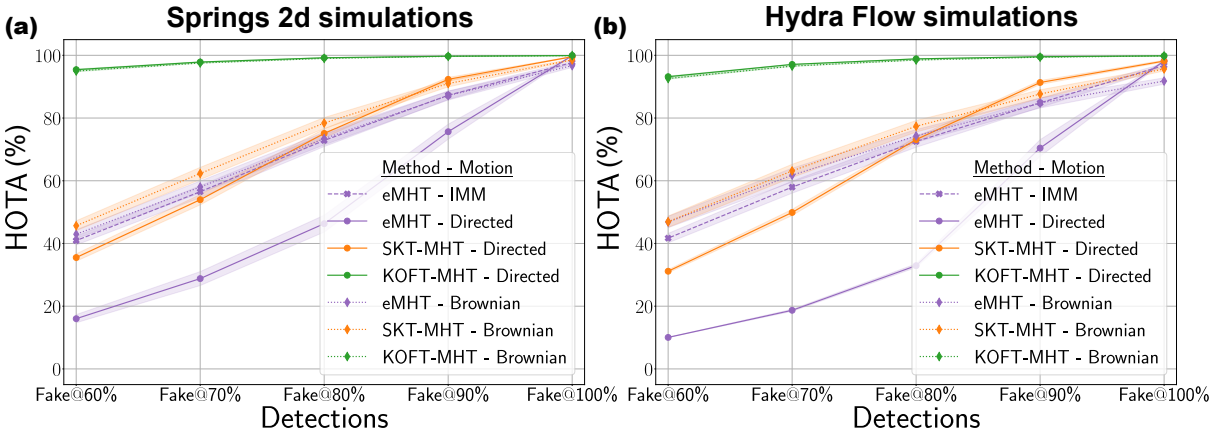


Figure A.1: Tracking performance of different motion models on synthetic videos. The HOTA metric [180] with 2 pixels tolerance is reported with respect to detection accuracy (Fake@F1). Methods are tested with different motion models: Brownian, Directed, or a mixture (IMM). The light-colored envelope around each curve corresponds to the 68% confidence interval (instead of 99.7% for visibility) over M = 5 simulations. **(a)** Results on Springs 2D synthetic videos **(b)** Results on Hydra Flow synthetic videos

many false or missed detections, Brownian motion is more robust. With a relatively shallow default tree depth ($\tau_{\text{depth}} = 3$), Directed motion may produce spurious tracks by consistently linking false positives that drift away from true targets. In contrast, Brownian motion, which assumes a null velocity, allows wrong associations to be corrected in subsequent frames.

Overall, our Bayesian tracker SKT-MHT consistently outperforms EMHT when both use the same motion model (Brownian or Directed). The IMM in EMHT provides a practical trade-off between both dynamics, at the cost of additional computational complexity, and has the advantage of requiring less prior knowledge from the user.

For KOFT-MHT, these considerations are less critical, thanks to the additional optical flow measurements that improve the robustness of both motion models. We note that Directed KOFT-MHT performs slightly better than its Brownian counterpart, although both models remain imperfect representations of the true motion. We expect that with a less accurate optical flow, the process model would become more important, allowing a larger gain of the Directed Kalman filter over the Brownian one.

Bibliography

[1] A. Hanson, "On being a hydra with, and without, a nervous system: what do neurons add?," *Animal Cognition*, vol. 26, no. 6, pp. 1799–1816, 2023. (See pages 3, 4, and 87)

[2] A. v. Leeuwenhoek, "Iv. part of a letter from mr antony van leeuwenhoek, frs concerning green weeds growing in water, and some animalcula found about them.," *Philosophical transactions of the Royal Society of London*, vol. 23, no. 283, pp. 1304–1311, 1702. (See page 2)

[3] A. Trembley, "Mémoires, pour servir à l'histoire d'un genre de polypes d'eau douce, à bras en forme de cornes," *(No Title)*, 1774. (See page 2)

[4] T. C. Bosch, "Why polyps regenerate and we don't: towards a cellular and molecular framework for hydra regeneration," *Developmental Biology*, vol. 303, no. 2, pp. 421–433, 2007. (See page 2)

[5] D. E. Martinez, "Mortality patterns suggest lack of senescence in hydra," *Experimental Gerontology*, vol. 33, no. 3, pp. 217–225, 1998. (See page 2)

[6] H. R. Bode, "Head regeneration in hydra," *Developmental Dynamics: An Official Publication of the American Association of Anatomists*, vol. 226, no. 2, pp. 225–236, 2003. (See page 2)

[7] T. Fujisawa, "Hydra regeneration and epitheliopeptides," *Developmental Dynamics: An Official Publication of the American Association of Anatomists*, vol. 226, no. 2, pp. 182–189, 2003. (See page 2)

[8] S. Han, E. Taralova, C. Dupre, and R. Yuste, "Comprehensive machine learning analysis of hydra behavior reveals a stable basal behavioral repertoire," *eLife*, vol. 7, pp. e32605, 2018. (See pages 3 and 4)

[9] S. Siebert, J. A. Farrell, J. F. Cazet, et al., "Stem cell differentiation trajectories in hydra resolved at single-cell resolution," *Science*, vol. 365, no. 6451, pp. eaav9314, 2019. (See page 4)

[10] C. Dupre and R. Yuste, "Non-overlapping neural networks in hydra vulgaris," *Current Biology*, vol. 27, no. 8, pp. 1085–1097, 2017. (See pages 4, 6, 7, 9, 13, 68, 72, 80, 86, 87, and 88)

[11] M. Chalfie, Y. Tu, G. Euskirchen, et al., "Green fluorescent protein as a marker for gene expression," *Science*, vol. 263, no. 5148, pp. 802–805, 1994. (See page 5)

[12] R. Y. Tsien, "The green fluorescent protein," *Annual Review of Biochemistry*, vol. 67, no. 1, pp. 509–544, 1998. (See page 5)

[13] M. Minsky, "Memoir on inventing the confocal scanning microscope," 1988. (See page 5)

[14] W. Denk, J. H. Strickler, and W. W. Webb, "Two-photon laser scanning fluorescence microscopy," *Science*, vol. 248, no. 4951, pp. 73–76, 1990. (See page 5)

[15] R. Yuste, "Fluorescence microscopy today," *Nature Methods*, vol. 2, no. 12, pp. 902–904, 2005. (See pages 5 and 70)

[16] P. Drapeau, D. W. Ali, R. R. Buss, and L. Saint-Amant, "In vivo recording from identifiable neurons of the locomotor network in the developing zebrafish," *Journal of Neuroscience Methods*, vol. 88, no. 1, pp. 1–13, 1999. (See page 6)

[17] M. B. Goodman, T. H. Lindsay, S. R. Lockery, and J. E. Richmond, "Electrophysiological methods for caenorhabditis elegans neurobiology," in *Methods in Cell Biology*, vol. 107, pp. 409–436. Elsevier, 2012. (See page 6)

[18] D. L. Gonzales, K. N. Badhiwala, D. G. Vercosa, et al., "Scalable electrophysiology in intact small animals with nanoscale suspended electrode arrays," *Nature Nanotechnology*, vol. 12, no. 7, pp. 684–691, 2017. (See page 6)

[19] J. Nakai, M. Ohkura, and K. Imoto, "A high signal-to-noise ca2+ probe composed of a single green fluorescent protein," *Nature Biotechnology*, vol. 19, no. 2, pp. 137–141, 2001. (See pages 6 and 70)

[20] M. H. Zhu, J. Jang, M. M. Milosevic, and S. D. Antic, "Population imaging discrepancies between a genetically-encoded calcium indicator (geci) versus a genetically-encoded voltage indicator (gevi)," *Scientific Reports*, vol. 11, no. 1, pp. 5295, 2021. (See pages 6 and 7)

[21] M. S. Siegel and E. Y. Isacoff, "A genetically encoded optical probe of membrane voltage," *Neuron*, vol. 19, no. 4, pp. 735–741, 1997. (See page 6)

[22] A. Hanson, R. Reme, N. Telerman, et al., "Automatic monitoring of neural activity with single-cell resolution in behaving hydra," *Scientific Reports*, vol. 14, no. 1, pp. 5083, 2024. (See pages 7, 8, 9, 11, 12, 13, 19, 20, 67, 68, 73, 80, 90, 125, 135, and 138)

[23] S. Kubler, *Statistical methods for the robust extraction of objects' spatio-temporal relations in bioimaging–Application to the functional analysis of neuronal networks in vivo*, Ph.D. thesis, Sorbonne Université, 2023. (See pages 9, 13, 14, and 143)

[24] J.-C. Olivo-Marin, "Extraction of spots in biological images using multiscale products," *Pattern Recognition*, vol. 35, no. 9, pp. 1989–1996, 2002. (See pages 10, 11, 17, 18, 27, 73, 87, 90, 92, 96, 117, 118, 121, 126, 131, 143, and 147)

[25] U. Schmidt, M. Weigert, C. Broaddus, and G. Myers, "Cell detection with star-convex polygons," in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2018: 21st international conference, Granada, Spain, September 16-20, 2018, proceedings, part II 11*. Springer, 2018, pp. 265–273. (See pages 10, 11, 17, 30, 64, 73, 143, and 144)

[26] N. Chenouard, I. Bloch, and J.-C. Olivo-Marin, "Multiple hypothesis tracking for cluttered biological image sequences," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2736–3750, 2013. (See pages 10, 43, 45, 46, 56, 64, 68, 73, 74, 79, 80, 90, 91, 92, 97, 99, 107, 113, 114, 116, 117, 118, 119, 126, 131, 137, and 147)

[27] K. Jaqaman, D. Loerke, M. Mettlen, et al., "Robust single-particle tracking in live-cell time-lapse sequences," *Nature Methods*, vol. 5, no. 8, pp. 695–702, 2008. (See pages 11, 16, 35, 37, 40, 41, 53, 64, 68, 73, 74, 80, 90, 91, 116, 125, 127, 130, 132, and 134)

[28] T. Lagache, A. Hanson, J. Pérez-Ortega, et al., "Tracking calcium dynamics from individual neurons in behaving animals," *PLoS Computational Biology*, vol. 17, pp. e1009432, 10 2021. (See pages 11, 18, 35, 67, 97, 125, 127, 132, 134, and 135)

[29] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955. (See pages 11 and 40)

[30] R. Jonker and T. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems," in *DGOR/NSOR: Papers of the 16th Annual Meeting of DGOR in Cooperation with NSOR/Vorträge der 16. Jahrestagung der DGOR zusammen mit der NSOR*. Springer, 1988, pp. 622–622. (See pages 11, 40, 84, 127, and 139)

[31] E. A. Pnevmatikakis, D. Soudry, Y. Gao, et al., "Simultaneous denoising, deconvolution, and demixing of calcium imaging data," *Neuron*, vol. 89, no. 2, pp. 285–299, 2016. (See pages 12 and 13)

[32] J. Friedrich and L. Paninski, "Fast active set methods for online spike inference from calcium imaging," *Advances in Neural Information Processing Systems*, vol. 29, 2016. (See pages 12 and 13)

[33] S. Kubler, S. Mukherjee, J.-C. Olivo-Marin, and T. Lagache, "A robust and versatile framework to compare spike detection methods in calcium imaging of neuronal activity," in *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2021, pp. 375–379. (See page 13)

[34] A. Giovannucci, J. Friedrich, P. Gunn, et al., "Caiman: An open source tool for scalable calcium imaging data analysis," *eLife*, vol. 8, pp. e38173, 2019. (See page 13)

[35] J.-M. Fellous, P. H. Tiesinga, P. J. Thomas, and T. J. Sejnowski, "Discovering spike patterns in neuronal responses," *Journal of Neuroscience*, vol. 24, no. 12, pp. 2989–3001, 2004. (See page 13)

[36] M. D. Humphries, "Spike-train communities: finding groups of similar spike trains," *Journal of Neuroscience*, vol. 31, no. 6, pp. 2321–2336, 2011. (See page 13)

[37] S. Kubler, J.-C. Olivo-Marin, and T. Lagache, "Statistical coupling between time point-processes," in *2022 IEEE 19th International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2022, pp. 1–4. (See page 13)

[38] J. K. Chapin and M. A. Nicolelis, "Principal component analysis of neuronal ensemble activity reveals multidimensional somatosensory representations," *Journal of Neuroscience Methods*, vol. 94, no. 1, pp. 121–140, 1999. (See page 13)

[39] V. Lopes-dos Santos, S. Conde-Ocazionez, M. A. Nicolelis, et al., "Neuronal assembly detection and cell membership specification by principal component analysis," *PloS One*, vol. 6, no. 6, pp. e20996, 2011. (See page 13)

[40] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, pp. P10008, 2008. (See page 13)

[41] G. Diana, T. T. Sainsbury, and M. P. Meyer, "Bayesian inference of neuronal assemblies," *PLoS Computational Biology*, vol. 15, no. 10, pp. e1007481, 2019. (See page 13)

[42] A. Mathis, P. Mamidanna, K. M. Cury, et al., "Deeplabcut: markerless pose estimation of user-defined body parts with deep learning," *Nature Neuroscience*, vol. 21, no. 9, pp. 1281–1289, 2018. (See page 14)

[43] T. D. Pereira, D. E. Aldarondo, L. Willmore, et al., "Fast animal pose estimation using deep neural networks," *Nature Methods*, vol. 16, no. 1, pp. 117–125, 2019. (See page 14)

[44] T. D. Pereira, N. Tabris, A. Matsliah, et al., "Sleap: A deep learning system for multi-animal pose tracking," *Nature Methods*, vol. 19, no. 4, pp. 486–495, 2022. (See page 14)

[45] W. Yamamoto and R. Yuste, "Peptide-driven control of somersaulting in hydra vulgaris," *Current Biology*, vol. 33, no. 10, pp. 1893–1905, 2023. (See pages 14 and 15)

[46] J. Mölter, L. Avitan, and G. J. Goodhill, "Detecting neural assemblies in calcium imaging data," *BMC Biology*, vol. 16, pp. 1–20, 2018. (See page 14)

[47] J. Pérez-Ortega, T. Alejandre-García, and R. Yuste, "Long-term stability of cortical ensembles," *eLife*, vol. 10, pp. e64449, 2021. (See page 14)

[48] A. Tampuu, T. Matiisen, H. F. Ólafsdóttir, et al., "Efficient neural decoding of self-location with a deep recurrent network," *PLoS Computational Biology*, vol. 15, no. 2, pp. e1006822, 2019. (See page 14)

[49] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. (See pages 14 and 43)

[50] F. de Chaumont, S. Dallongeville, N. Chenouard, et al., "Icy: an open bioimage informatics platform for extended reproducible research," *Nature Methods*, vol. 9, no. 7, pp. 690–696, 2012. (See pages 15, 46, 64, 73, 74, 90, 91, 92, 117, and 126)

[51] J. Schindelin, I. Arganda-Carreras, E. Frise, et al., "Fiji: an open-source platform for biological-image analysis," *Nature Methods*, vol. 9, no. 7, pp. 676–682, 2012. (See pages 15, 41, 64, 73, 74, 90, and 91)

[52] N. Sofroniew, T. Lambert, G. Bokota, et al., "napari: a multi-dimensional image viewer for python," 2025. (See page 15)

[53] J. Huth, M. Buchholz, J. M. Kraus, et al., "Significantly improved precision of cell migration analysis in time-lapse video microscopy through use of a fully automated tracking system," *BMC Cell Biology*, vol. 11, pp. 1–12, 2010. (See pages 16, 37, 41, 42, and 53)

[54] N. Chenouard, I. Smal, F. de Chaumont, et al., "Objective comparison of particle tracking methods," *Nature Methods*, vol. 11, no. 3, pp. 281–289, 2014. (See pages 16 and 17)

[55] D. Svoboda and V. Ulman, "Mitogen: a framework for generating 3d synthetic time-lapse sequences of cell populations in fluorescence microscopy," *IEEE transactions on Medical Imaging*, vol. 36, no. 1, pp. 310–321, 2016. (See pages 16 and 67)

[56] I. Smal and E. Meijering, "Quantitative comparison of multiframe data association techniques for particle tracking in time-lapse fluorescence microscopy," *Medical Image Analysis*, vol. 24, no. 1, pp. 163–189, 2015. (See page 16)

[57] A. Genovesio, T. Liedl, V. Emiliani, et al., "Multiple particle tracking in 3-d+ t microscopy: method and application to the tracking of endocytosed quantum dots," *IEEE Transactions on Image Processing*, vol. 15, no. 5, pp. 1062–1070, 2006. (See pages 16, 37, 41, 47, 53, 56, 79, 80, 84, 97, and 117)

[58] I. Smal, K. Draegestein, N. Galjart, et al., "Particle filtering for multiple object tracking in dynamic fluorescence microscopy images: Application to microtubule growth analysis," *IEEE Transactions on Medical Imaging*, vol. 27, no. 6, pp. 789–804, 2008. (See pages 16, 54, 58, and 101)

[59] M. Maška, V. Ulman, D. Svoboda, et al., "A benchmark for comparison of cell tracking algorithms," *Bioinformatics*, vol. 30, no. 11, pp. 1609–1617, 2014. (See pages 16, 67, and 138)

[60] T. Meinhardt, A. Kirillov, L. Leal-Taixe, and C. Feichtenhofer, "Trackformer: Multi-object tracking with transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8844–8854. (See pages 16 and 49)

[61] M. Maška, V. Ulman, P. Delgado-Rodriguez, et al., "The cell tracking challenge: 10 years of objective benchmarking," *Nature Methods*, vol. 20, no. 7, pp. 1010–1020, 2023. (See pages 16, 17, and 67)

[62] C. Malin-Mayor, P. Hirsch, L. Guignard, et al., "Automated reconstruction of whole-embryo cell lineages by learning from sparse annotations," *Nature Biotechnology*, vol. 41, no. 1, pp. 44–49, 2023. (See pages 16, 30, 32, 34, 35, 49, and 147)

[63] K. Löffler and R. Mikut, "Embedtrack—simultaneous cell segmentation and tracking through learning offsets and clustering bandwidths," *IEEE Access*, vol. 10, pp. 77147–77157, 2022. (See pages 16, 37, 47, and 143)

[64] B. Gallusser and M. Weigert, "Trackastra: Transformer-based cell tracking for live-cell microscopy," in *European Conference on Computer Vision*. Springer, 2025, pp. 467–484. (See pages 16, 30, 32, 34, 35, and 140)

[65] R. Reme, A. Newson, E. Angelini, et al., "Sinetra: A versatile framework for evaluating single neuron tracking in behaving animals," in *2025 IEEE 22nd International Symposium on Biomedical Imaging (ISBI)*, 2025, pp. 1–5. (See pages 17, 19, 20, 67, 86, 90, 93, 96, 97, and 118)

[66] R. Reme, A. Newson, E. Angelini, et al., "Particle tracking in biological images with optical-flow enhanced kalman filtering," in *2024 IEEE International Symposium on Biomedical Imaging (ISBI)*, 2024, pp. 1–5. (See pages 18, 19, 20, 79, and 140)

[67] R. Reme, V. Piriou, A. Hanson, et al., "Tracking intermittent particles with self-learned visual features," in *2023 IEEE 20th International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2023, pp. 1–5. (See pages 19, 20, 35, and 125)

[68] S. Bonneau, M. Dahan, and L. D. Cohen, "Single quantum dot tracking based on perceptual grouping using minimal paths in a spatiotemporal volume," *IEEE Transactions on Image Processing*, vol. 14, no. 9, pp. 1384–1395, 2005. (See pages 22 and 125)

[69] M. J. Rust, M. Bates, and X. Zhuang, "Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (storm)," *Nature Methods*, vol. 3, no. 10, pp. 793–796, 2006. (See pages 22 and 125)

[70] D. Smetters, A. Majewska, and R. Yuste, "Detecting action potentials in neuronal populations with calcium imaging," *Methods*, vol. 18, no. 2, pp. 215–221, 1999. (See page 22)

[71] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: nonlinear phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992. (See page 27)

[72] A. Chambolle, "An algorithm for total variation minimization and applications," *Journal of Mathematical Imaging and Vision*, vol. 20, pp. 89–97, 2004. (See page 27)

[73] T. Lindeberg, "Scale-space theory: A basic tool for analyzing structures at different scales," *Journal of Applied Statistics*, vol. 21, no. 1-2, pp. 225–270, 1994. (See page 27)

[74] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the seventh IEEE International Conference on Computer Vision*. Ieee, 1999, vol. 2, pp. 1150–1157. (See page 27)

[75] N. Otsu et al., "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, no. 285-296, pp. 23–27, 1975. (See page 27)

[76] P.-S. Liao, T.-S. Chen, P.-C. Chung, et al., "A fast algorithm for multilevel thresholding," *J. Inf. Sci. Eng.*, vol. 17, no. 5, pp. 713–727, 2001. (See page 27)

[77] S. Beucher, "Use of watersheds in contour detection," in *Proc. Int. Workshop on Image Processing, Sept. 1979*, 1979, pp. 17–21. (See page 28)

[78] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988. (See pages 28 and 48)

[79] R. Goldenberg, R. Kimmel, E. Rivlin, and M. Rudzsky, "Fast geodesic active contours," *IEEE Transactions on Image Processing*, vol. 10, no. 10, pp. 1467–1475, 2001. (See page 28)

[80] A. Dufour, V. Meas-Yedid, A. Grassart, and J.-C. Olivo-Marin, "Automated quantification of cell endocytosis using active contours and wavelets," in *2008 19th International Conference on Pattern Recognition*. IEEE, 2008, pp. 1–4. (See page 28)

[81] A. Badoual, D. Schmitter, V. Uhlmann, and M. Unser, "Multiresolution subdivision snakes," *IEEE Transactions on Image Processing*, vol. 26, no. 3, pp. 1188–1201, 2016. (See page 28)

[82] A. Habis, V. Meas-Yedid, E. Angelini, and J.-C. Olivo-Marin, "Deep contourflow: Advancing active contours with deep learning," *arXiv preprint arXiv:2407.10696*, 2024. (See page 28)

[83] S. Osher and J. A. Sethian, "Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations," *Journal of Computational Physics*, vol. 79, no. 1, pp. 12–49, 1988. (See page 28)

[84] T. F. Chan and L. A. Vese, "Active contours without edges," *IEEE Transactions on Image Processing*, vol. 10, no. 2, pp. 266–277, 2001. (See pages 28 and 48)

[85] C. Zimmer and J.-C. Olivo-Marin, "Coupled parametric active contours," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 11, pp. 1838–1842, 2005. (See pages 28 and 48)

[86] A. Dufour, R. Thibeaux, E. Labruyere, et al., "3-d active meshes: Fast discrete deformable models for cell tracking in 3-d time-lapse microscopy," *IEEE Transactions on Image Processing*, vol. 20, no. 7, pp. 1925–1937, 2011. (See pages 28, 29, and 48)

[87] D. B. Mumford and J. Shah, "Optimal approximations by piecewise smooth functions and associated variational problems," *Communications on Pure and Applied Mathematics*, 1989. (See pages 28 and 48)

[88] A. Baddeley and M. V. Lieshout, "Stochastic geometry models in high-level vision," *Journal of Applied Statistics*, vol. 20, no. 5-6, pp. 231–256, 1993. (See page 29)

[89] H. Rue and M. A. Hurn, "Bayesian object identification," *Biometrika*, vol. 86, no. 3, pp. 649–660, 1999. (See page 29)

[90] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," *Advances in Neural Information Processing Systems*, vol. 24, 2011. (See page 29)

[91] E. Türetken, X. Wang, C. J. Becker, et al., "Network flow integer programming to track elliptical cells in time-lapse sequences," *IEEE Transactions on Medical Imaging*, vol. 36, no. 4, pp. 942–951, 2017. (See pages 29, 30, 32, 33, 35, and 143)

[92] J. Mabon, M. Ortner, and J. Zerubia, "Learning point processes and convolutional neural networks for object detection in satellite images," *Remote Sensing*, vol. 16, no. 6, pp. 1019, 2024. (See page 29)

[93] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587. (See page 29)

[94] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1440–1448. (See page 29)

[95] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in Neural Information Processing Systems*, vol. 28, 2015. (See page 29)

[96] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788. (See page 29)

[97] W. Liu, D. Anguelov, D. Erhan, et al., "Ssd: Single shot multibox detector," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, 2016, pp. 21–37. (See page 29)

[98] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. (See pages 29 and 47)

[99] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer, 2015, pp. 234–241. (See pages 29 and 34)

[100] C. Stringer, T. Wang, M. Michaelos, and M. Pachitariu, "Cellpose: a generalist algorithm for cellular segmentation," *Nature Methods*, vol. 18, no. 1, pp. 100–106, 2021. (See page 30)

[101] K. J. Cutler, C. Stringer, T. W. Lo, et al., "Omnipose: a high-precision morphology-independent solution for bacterial cell segmentation," *Nature Methods*, vol. 19, no. 11, pp. 1438–1448, 2022. (See page 30)

[102] A. D. Mantes, A. Herrera, I. Khven, et al., "Spotiflow: accurate and efficient spot detection for fluorescence microscopy with deep stereographic flow regression," *bioRxiv*, pp. 2024–02, 2024. (See page 30)

[103] A. Dosovitskiy, L. Beyer, A. Kolesnikov, et al., "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020. (See page 30)

[104] N. Carion, F. Massa, G. Synnaeve, et al., "End-to-end object detection with transformers," in *European Conference on Computer Vision*. Springer, 2020, pp. 213–229. (See page 30)

[105] D. Doncila Pop, P. Le Bodic, and J. Nunez-Iglesias, "Needles in the haystack - rapid error detection for cell tracking solutions," in *2025 IEEE 22nd International Symposium on Biomedical Imaging*. 2025, IEEE. (See pages 30, 32, 33, 35, 140, 143, and 144)

[106] T. Ben-Haim and T. R. Raviv, "Graph neural network for cell tracking in microscopy videos," in *European Conference on Computer Vision*. Springer, 2022, pp. 610–626. (See pages 30, 34, and 35)

[107] J. Pineda, B. Midtvedt, H. Bachimanchi, et al., "Geometric deep learning reveals the spatiotemporal features of microscopic motion," *Nature Machine Intelligence*, vol. 5, no. 1, pp. 71–82, 2023. (See pages 30, 34, 35, and 36)

[108] L. A. Wolsey and G. L. Nemhauser, *Integer and combinatorial optimization*, John Wiley & Sons, 1999. (See pages 33 and 46)

[109] Q. Huangfu and J. J. Hall, "Parallelizing the dual revised simplex method," *Mathematical Programming Computation*, vol. 10, no. 1, pp. 119–142, 2018. (See pages 33, 46, and 110)

[110] T. Achterberg, "Scip: solving constraint integer programs," *Mathematical Programming Computation*, vol. 1, no. 1, pp. 1–41, Jul 2009. (See pages 33 and 46)

[111] Gurobi Optimization, LLC, *Gurobi optimizer reference manual*, 2021. (See pages 33, 46, 47, and 110)

[112] J. H. Friedman, "Stochastic gradient boosting," *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378, 2002. (See page 33)

[113] A. Vaswani, N. Shazeer, N. Parmar, et al., "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017. (See page 34)

[114] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013. (See pages 34 and 43)

[115] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778. (See pages 34, 131, and 134)

[116] M. Schiegg, P. Hanslovsky, B. X. Kausler, et al., "Conservation tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2928–2935. (See pages 35 and 143)

[117] C. Haubold, M. Schiegg, A. Kreshuk, et al., "Segmenting and tracking multiple dividing targets using ilastik," *Focus on Bio-Image Informatics*, pp. 199–229, 2016. (See pages 35 and 143)

[118] D. Sage, F. Neumann, F. Hediger, et al., "Automatic tracking of individual fluorescence particles: application to the study of chromosome dynamics," *IEEE Transactions on Image Processing*, vol. 14, no. 9, pp. 1372–1383, 2005. (See page 35)

[119] K. E. Magnusson, J. Jaldén, P. M. Gilbert, and H. M. Blau, "Global linking of cell tracks using the viterbi algorithm," *IEEE transactions on medical imaging*, vol. 34, no. 4, pp. 911–929, 2014. (See pages 35, 61, and 140)

[120] K. E. G. Magnusson and J. Jaldén, "Tracking of non-brownian particles using the viterbi algorithm," in *2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI)*, 2015, pp. 380–384. (See pages 35 and 37)

[121] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967. (See page 36)

[122] A. Genovesio, B. Zhang, and J.-C. Olivo-Marin, "Tracking of multiple fluorescent biological objects in three dimensional video microscopy," in *Proceedings 2003 International Conference on Image Processing (Cat. No. 03CH37429)*. IEEE, 2003, vol. 1, pp. I–1105. (See pages 37 and 53)

[123] I. F. Sbalzarini and P. Koumoutsakos, "Feature point tracking and trajectory analysis for video imaging in cell biology," *Journal of Structural Biology*, vol. 151, no. 2, pp. 182–195, 2005. (See pages 37, 41, 53, and 130)

[124] K. Ulicna, G. Vallardi, G. Charras, and A. R. Lowe, "Automated deep lineage tree analysis using a bayesian single cell tracking approach," *Frontiers in Computer Science*, vol. 3, pp. 734559, 2021. (See pages 37, 42, 79, and 80)

[125] Y. Yao, I. Smal, and E. Meijering, "Deep neural networks for data association in particle tracking," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*. IEEE, 2018, pp. 458–461. (See pages 37 and 40)

[126] E. Moen, E. Borba, G. Miller, et al., "Accurate cell tracking and lineage construction in live-cell imaging experiments with deep learning," *Biorxiv*, p. 803205, 2019. (See pages 37, 40, 43, and 64)

[127] A. Genovesio, B. Zhang, and J.-C. Olivo-Marin, "Interacting multiple model based method to track moving fluorescent biological spots," in *2004 2nd IEEE International Symposium on Biomedical Imaging: Nano to Macro (IEEE Cat No. 04EX821)*. IEEE, 2004, pp. 1239–1242. (See pages 41 and 56)

[128] J.-Y. Tinevez, N. Perry, J. Schindelin, et al., "Trackmate: An open and extensible platform for single-particle tracking," *Methods*, vol. 115, pp. 80–90, 2017. (See pages 41, 43, 64, 73, 74, 79, 80, 90, 91, 116, and 140)

[129] C. Morefield, "Application of 0-1 integer programming to multitarget tracking problems," *IEEE Transactions on Automatic Control*, vol. 22, no. 3, pp. 302–312, 1977. (See pages 43, 99, and 107)

[130] D. Reid, "An algorithm for tracking multiple targets," *IEEE Transactions on Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979. (See pages 43, 45, 46, 59, 105, and 107)

[131] A. B. Poore, "Multidimensional assignment formulation of data association problems arising from multitarget and multisensor tracking," *Computational Optimization and Applications*, vol. 3, no. 1, pp. 27–57, 1994. (See pages 43, 99, and 107)

[132] L. Feng, Y. Xu, Y. Yang, and X. Zheng, "Multiple dense particle tracking in fluorescence microscopy images based on multidimensional assignment," *Journal of Structural Biology*, vol. 173, no. 2, pp. 219–228, 2011. (See pages 43, 45, 47, 56, 79, 80, 99, 107, 116, 117, 118, 119, 140, and 144)

[133] Y. Yao, I. Smal, I. Grigoriev, et al., "Deep-learning method for data association in particle tracking," *Bioinformatics*, vol. 36, no. 19, pp. 4935–4941, 2020. (See pages 43, 45, 47, 64, and 107)

[134] T. Kaiser, M. Schier, and B. Rosenhahn, "Cell tracking according to biological needs–strong mitosis-aware multi-hypothesis tracker with aleatoric uncertainty," *arXiv preprint arXiv:2403.15011*, 2024. (See pages 43, 47, 99, 107, 140, and 144)

[135] K. G. Murty, "An algorithm for ranking all the assignments in order of increasing cost," *Operations Research*, vol. 16, no. 3, pp. 682–687, 1968. (See page 47)

[136] M. L. Miller, H. S. Stone, and I. J. Cox, "Optimizing murty's ranked assignment method," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 33, no. 3, pp. 851–862, 1997. (See page 47)

[137] D. Guo, A. L. Van de Ven, and X. Zhou, "Tracking and measurement of the motion of blood cells using optical flow methods," *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 3, pp. 991, 2014. (See pages 48, 84, and 147)

[138] A. Boquet-Pujadas and J.-C. Olivo-Marin, "Reformulating optical flow to solve image-based inverse problems and quantify uncertainty," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 5, pp. 6125–6141, 2022. (See pages 48 and 84)

[139] Y. Kesenci, A. Boquet-Pujadas, M. Unser, and J.-C. Olivo-Marin, "Estimation of stiffness maps in deforming cells through optical flow with bounded curvature," *IEEE Transactions on Medical Imaging*, 2024. (See pages 48 and 84)

[140] G. Balakrishnan, A. Zhao, M. R. Sabuncu, et al., "Voxelmorph: a learning framework for deformable medical image registration," *IEEE Transactions on Medical Imaging*, vol. 38, no. 8, pp. 1788–1800, 2019. (See pages 48, 49, 84, 87, and 88)

[141] T. Senst, V. Eiselein, and T. Sikora, "Robust local optical flow for feature tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 9, pp. 1377–1387, 2012. (See pages 48 and 84)

[142] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981. (See pages 49, 80, and 84)

[143] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime tv-l 1 optical flow," in *Pattern Recognition: 29th DAGM Symposium*. Springer, 2007, pp. 214–223. (See pages 49, 84, 87, and 88)

[144] J. Sánchez Pérez, E. Meinhardt-Llopis, and G. Facciolo, "TV-L1 Optical Flow Estimation," *Image Processing On Line*, vol. 3, pp. 137–150, 2013, `https://doi.org/10.5201/ipol.2013.26`. (See pages 49, 87, and 88)

[145] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Computer Vision-ECCV 2004: 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004. Proceedings, Part IV 8*. Springer, 2004, pp. 25–36. (See page 49)

[146] E. Meinhardt-Llopis and J. Sánchez, "Horn-schunck optical flow with a multi-scale strategy," *Image Processing On Line*, 2013. (See page 49)

[147] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *Computer Vision–ECCV 2020: 16th European Conference*. Springer, 2020, pp. 402–419. (See pages 49, 84, 87, and 88)

[148] F. Boukari and S. Makrogiannis, "Automated cell tracking using motion prediction-based matching and event handling," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 17, no. 3, pp. 959–971, 2020. (See pages 49 and 147)

[149] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2015. (See page 49)

[150] L. Bertinetto, J. Valmadre, J. F. Henriques, et al., "Fully-convolutional siamese networks for object tracking," in *Computer vision–ECCV 2016 workshops: Amsterdam, the Netherlands, October 8-10 and 15-16, 2016, proceedings, part II 14*. Springer, 2016, pp. 850–865. (See page 49)

[151] B. Li, J. Yan, W. Wu, et al., "High performance visual tracking with siamese region proposal network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8971–8980. (See page 49)

[152] Y. Xu, Z. Wang, Z. Li, et al., "Siamfc++: Towards robust and accurate visual tracking with target estimation guidelines," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020, vol. 34, pp. 12549–12556. (See page 49)

[153] J. Ryu, A. Nejatbakhsh, M. Torkashvand, et al., "Versatile multiple object tracking in sparse 2d/3d videos via deformable image registration," *PLOS Computational Biology*, vol. 20, no. 5, pp. e1012075, 2024. (See pages 49, 67, 68, 73, 74, and 84)

[154] P. Bergmann, T. Meinhardt, and L. Leal-Taixe, "Tracking without bells and whistles," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 941–951. (See page 49)

[155] F. Zeng, B. Dong, Y. Zhang, et al., "Motr: End-to-end multiple-object tracking with transformer," in *European Conference on Computer Vision*. Springer, 2022, pp. 659–675. (See page 49)

[156] E. Yu, T. Wang, Z. Li, et al., "Motrv3: Release-fetch supervision for end-to-end multi-object tracking," *arXiv preprint arXiv:2305.14298*, 2023. (See page 49)

[157] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 03 1960. (See page 51)

[158] G. Bishop, G. Welch, et al., "An introduction to the kalman filter," *Proc of SIGGRAPH, Course*, vol. 8, no. 27599-23175, pp. 41, 2001. (See page 51)

[159] A. H. Jazwinski, *Stochastic processes and filtering theory*, Courier Corporation, 2007. (See pages 51 and 54)

[160] S. J. Julier and J. K. Uhlmann, "New extension of the kalman filter to nonlinear systems," in *Signal Processing, Sensor Fusion, and Target Recognition VI*. Spie, 1997, vol. 3068, pp. 182–193. (See pages 51 and 54)

[161] N. J. Gordon, D. J. Salmond, and A. F. Smith, "Novel approach to nonlinear/non-gaussian bayesian state estimation," in *IEE proceedings F (Radar and Signal Processing)*. IET, 1993, vol. 140, pp. 107–113. (See pages 51 and 54)

[162] R. Labbe, "Kalman and bayesian filters in python," `https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python`, 9e3d2f6ed, 2014. (See pages 51 and 53)

[163] A. Doucet, N. De Freitas, N. J. Gordon, et al., *Sequential Monte Carlo methods in practice*, vol. 1, Springer, 2001. (See page 54)

[164] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002. (See page 54)

[165] I. Smal, W. Niessen, and E. Meijering, "Advanced particle filtering for multiple object tracking in dynamic fluorescence microscopy images," in *2007 4th IEEE International Symposium on Biomedical Imaging: From Nano to Macro*. IEEE, 2007, pp. 1048–1051. (See pages 54 and 58)

[166] I. Smal, E. Meijering, K. Draegestein, et al., "Multiple object tracking in molecular bioimaging by rao-blackwellized marginal particle filtering," *Medical Image Analysis*, vol. 12, no. 6, pp. 764–777, 2008. (See pages 54 and 58)

[167] H. A. Blom and Y. Bar-Shalom, "The interacting multiple model algorithm for systems with markovian switching coefficients," *IEEE Transactions on Automatic Control*, vol. 33, no. 8, pp. 780–783, 2002. (See page 55)

[168] H. E. Rauch, F. Tung, and C. T. Striebel, "Maximum likelihood estimates of linear dynamic systems," *AIAA Journal*, vol. 3, no. 8, pp. 1445–1450, 1965. (See pages 57 and 86)

[169] R. Mahler, *Statistical multisource-multitarget information fusion*, Artech, 2007. (See pages 59 and 60)

[170] R. P. Mahler, *Advances in Statistical Multisource-Multitarget Information Fusion*, Artech House, 2014. (See pages 59 and 60)

[171] R. P. Mahler, "Multitarget bayes filtering via first-order multitarget moments," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1152–1178, 2004. (See page 61)

[172] B.-N. Vo and W.-K. Ma, "The gaussian mixture probability hypothesis density filter," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4091–4104, 2006. (See page 61)

[173] B.-T. Vo and B.-N. Vo, "Labeled random finite sets and multi-object conjugate priors," *IEEE Transactions on Signal Processing*, vol. 61, no. 13, pp. 3460–3475, 2013. (See page 61)

[174] B.-N. Vo, B.-T. Vo, and D. Phung, "Labeled random finite sets and the bayes multi-target tracking filter," *IEEE Transactions on Signal Processing*, vol. 62, no. 24, pp. 6554–6567, 2014. (See pages 61 and 62)

[175] A. Matyjaszkiewicz, G. Fiore, F. Annunziata, et al., "Bsim 2.0: an advanced agent-based cell simulator," *ACS Synthetic Biology*, vol. 6, no. 10, pp. 1969–1972, 2017. (See page 67)

[176] V. Ulman, M. Maška, K. E. Magnusson, et al., "An objective comparison of cell-tracking algorithms," *Nature Methods*, vol. 14, no. 12, pp. 1141–1152, 2017. (See page 67)

[177] C. F. Park, M. Barzegar-Keshteli, K. Korchagina, et al., "Automated neuron tracking inside moving and deforming c. elegans using deep learning and targeted augmentation," *Nature Methods*, vol. 21, no. 1, pp. 142–149, 2024. (See page 67)

[178] X. Yu, M. S. Creamer, F. Randi, et al., "Fast deep neural correspondence for tracking and identifying neurons in c. elegans using semi-synthetic training," *eLife*, vol. 10, pp. e66410, 2021. (See page 67)

[179] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," in *Image Analysis: 13th Scandinavian Conference, SCIA*. Springer, 2003, pp. 363–370. (See pages 70, 84, 87, 88, 91, 95, 96, 143, and 147)

[180] J. Luiten, A. Osep, P. Dendorfer, et al., "Hota: A higher order metric for evaluating multi-object tracking," *International Journal of Computer Vision*, vol. 129, pp. 548–578, 2021. (See pages 73, 74, 89, 91, 95, 117, 119, 120, 121, 147, and 149)

[181] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000. (See pages 88 and 91)

[182] J. P. Nguyen, A. N. Linder, G. S. Plummer, et al., "Automatically tracking neurons in a moving and deforming brain," *PLoS Computational Biology*, vol. 13, no. 5, pp. e1005517, 2017. (See page 97)

[183] P. Virtanen, R. Gommers, T. E. Oliphant, et al., "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020. (See page 110)

[184] G. C. Chow, *Analysis and control of dynamic economic systems*, Wiley, 1975. (See page 115)

[185] T. M. Inc., "Matlab version: 9.13.0 (r2022b)," 2022. (See page 117)

[186] N. Chenouard, J.-C. Olivo-Marin, et al., "Feature-aided particle tracking," in *2008 15th IEEE International Conference on Image Processing*. IEEE, 2008, pp. 1796–1799. (See pages 130 and 134)

[187] N. Chenouard, F. de Chaumont, I. Bloch, and J.-C. Olivo-Marin, "Improving 3d tracking in microscopy by joint estimation of kinetic and image models," in *MICCAI Workshop on MIAAB*. Citeseer, 2008. (See page 130)

[188] J. Wang, Y. Song, T. Leung, et al., "Learning fine-grained image similarity with deep ranking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1386–1393. (See pages 130 and 134)

[189] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. (See page 131)

[190] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017. (See page 131)

[191] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018. (See page 134)

[192] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International Conference on Machine Learning*. PmLR, 2020, pp. 1597–1607. (See page 134)

[193] J. Zbontar, L. Jing, I. Misra, et al., "Barlow twins: Self-supervised learning via redundancy reduction," in *International Conference on Machine Learning*. PMLR, 2021, pp. 12310–12320. (See page 134)

[194] P. Matula, M. Maška, D. V. Sorokin, et al., "Cell tracking accuracy measurement based on comparison of acyclic oriented graphs," *PloS One*, vol. 10, no. 12, pp. e0144959, 2015. (See page 138)

[195] M. Manich, P. Bochet, A. Boquet-Pujadas, et al., "Fibronectin induces a transition from amoeboid to a fan morphology and modifies migration in entamoeba histolytica," *PLoS Pathogens*, vol. 20, no. 7, pp. e1012392, 2024. (See page 141)

[196] G. Nardi, M. S. Sano, M. Bilay, et al., "Characterizing particle dynamics in live imaging through stochastic physical models and machine learning," *bioRxiv*, pp. 2024–12, 2024. (See page 141)

[197] A. Genovesio and J.-C. Olivo-Marin, "Split and merge data association filter for dense multi-target tracking," in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* IEEE, 2004, vol. 4, pp. 677–680. (See page 143)

[198] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2017, pp. 3645–3649. (See page 144)

[199] Y. Du, Z. Zhao, Y. Song, et al., "Strongsort: Make deepsort great again," *IEEE Transactions on Multimedia*, 2023. (See page 144)

[200] K. Liu, S. S. Lienkamp, A. Shindo, et al., "Optical flow guided cell segmentation and tracking in developing tissue," in *2014 IEEE 11th International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2014, pp. 298–301. (See page 147)