Gifu University

Doctoral Dissertation

Modelling and Simulation of Biosensors Driven by Myosin Motors

モーター蛋白質ミオシンによって駆動されるバイオセンサーのモデリングとシミュレーション

Doctor of Philosophy in Engineering
Electronics and Information Systems Engineering Division
·Biosensors · Biomechanics · Biomechatronics · Simulation

Samuel Macharia Kang'iri

Year 2022

GIFU UNIVERSITY

DOCTORAL DISSERTATION

Modelling and Simulation of Biosensors Driven by Myosin Motors

モーター蛋白質ミオシンによって駆動されるバイオセンサーのモデリングとシミュレーション

Author:
Samuel Macharia Kang'iri

Supervisors:
Takahiro Nitta and
Minoru Sasaki

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Award of Degree of
Doctor of Philosophy in Engineering,
Electronics and Information Systems Engineering Division,

in the

Graduate School of Engineering,
Gifu University, Japan.

岐阜大学
GIFU UNIVERSITY

June 2, 2022

DECLARATION

I, Samuel Macharia Kang'iri (Sam Macharia), declare that this dissertation is original work and, to the best of my knowledge, has not been presented in any university or institution for a degree, or for consideration of any certification, and that external sources used here have been acknowledged in references.

Signature .... サム Date 3rd June, 2022 ....
Samuel Macharia Kang'iri
Reg. No. 1183914101
Electronics and Information Systems Engineering Division,
Graduate School of Engineering,
Gifu University, Japan.

Approval

This dissertation has been submitted with my approval as the supervisor.

Signature .... Date ....
Prof. Takahiro Nitta (Ph.D.)
Department of Electrical, Electronic and Computer Engineering,
Applied Physics Course,
Gifu University, Japan.

***

This research was carried out in Prof. Takahiro Nitta Lab. Applied Physics course, Electronics and Information Systems Engineering Division, Department of Electrical, Electronics, and Computer Engineering, Graduate School of Engineering, Gifu University, Gifu City, Japan.

DEDICATION

To engineers who seek deeper insights	English
より深い洞察を求めるエンジニアへ	Japanese
Kwa wahandisi wanaotafuta maarifa ya kina	Swahili
Kũrĩ iijinia arĩa macaragia ũũgĩ mũrikĩru	Kikuyu

***

"In the fields of observation, chance favours only the prepared mind."

Lecture, University of Lille
LOUIS PASTEUR, DECEMBER 7, 1854

ACKNOWLEDGEMENT

I express my sincere gratitude to all my supervisors, Prof. Takahiro Nitta , Prof. Minoru Sasaki , and Prof. Shintaro Kondo (Gifu University, Gifu, Japan), and Prof. Dan V. Nicolau (McGill University, Montreal, Canada). Their vast knowledge, constructive criticism, and instructional support were invaluable in my Ph.D. studies.

I am infinitely grateful to my family and friends for taking interest in my Ph.D. studies. Without your appreciation, empathy, encouragement, and trust, I could not pursue my Ph.D. with enthusiasm.

I also thank Dedan Kimathi University of Technology, Kenya and Gifu University, Japan, for the productive collaboration that made this research possible.

***

An acknowledgement may be susceptible to *fundamental attribution error*.

ABSTRACT

Mechanical movement of animals is achieved by muscle contraction and relaxation. There are three types of muscles responsible for different movements in the body. First, skeletal muscles are attached to the bones and are responsible for body movements such as locomotion and pick-and-place tasks. Cardiac muscles are found in the heart and are essential for pumping blood throughout the body as needed. Smooth muscles are found in the trachea, intestinal wall, arteries, and veins and help move body fluids and materials. All of these muscles are driven by actin and myosin. Both skeletal and cardiac muscles are called striated muscles because they have a sarcomere, a repeated streak of actin and myosin. However, smooth muscles do not have the sarcomere and operate involuntarily. Underneath the smooth and striated muscle contraction and relaxation mechanism, the myosin biomolecular motors slide onto the actin filaments to allow the muscle mechanism to work. To elucidate the detailed working principle of these biomolecular motor systems, they can be extracted from muscle cells and used to perform *in vitro* motility assays. These motility assays are the basis for various micro-nano devices such as biosensors and biocomputers. Biomolecular motors are small in size, have high speed, low energy requirements, and high energy conversion efficiency. In the current state of engineering, it is still difficult to construct an artificial motor that can match the combined properties of these biomolecular motors. Understanding biomolecular motors is indispensable in physics and engineering. With this understanding, we can achieve biocompatible high-speed material transportation and control in micronanoscale engineering devices without constructing artificial nanoscale motors. Furthermore, it is easier to power devices based on biomolecular motors because only adenosine triphosphate (ATP) is required as a source of energy. ATP is ubiquitous and its chemical energy can be converted directly to mechanical energy (power stroke), and unlike conventional motors, heating is not involved. Myosin biomolecular motors and their associated actin filaments are the main focus of this study. Although myosin-driven devices generally have higher translation speeds, there are two main problems involved. First, when the *in vitro* motility assay is performed, some myosins become defective and need to be removed from the motility system because they introduce impedance. Second, actin filaments have high flexibility, which decreases the efficiency of cargo transport or drug delivery to the desired location due to their inherent intermittent wiggle. In the first problem, defective myosins bind to the actin filament and resist forward propulsion by staying stuck to the actin filament. It is difficult to quantitatively investigate the effects of these defective binding myosin in *in vitro* motility assay experiments. This difficulty arises from the inability to set the active ratio, the irreproducibility of an experiment with *in vitro* motility, and the spatiotemporal resolution. Instead, using computer simulation, we can set the active ratio and other parameters as needed. We can also simulate various motility assays with high reproducibility of the results. We can then iterate the simulation and analysis process to obtain testable results comparable to peer-reviewed experimental results. The simulation results are noteworthy because they can reveal a deeper understanding before the actual experiment is performed. These insights from the simulation results improve the preparation of the experiments and build intuition of the experimental results. In this study, by developing a 3D Brownian dynamics simulation, we studied the effects of introducing these defective motors into an active actin-myosin system. We modelled the actin filament as a chain of rigid (inextensible) rods connected by beads. These rigid rods can bend around the beads. The assemblage of the rigid rods and beads makes a semi-flexible bead-rod polymer model equivalent to the mechanical properties of the actin filament. The myosin motors were modelled as linear springs with fixed spring constants equivalent to the experimental studies. To make the motility assay in silico, we generated these myosin motors randomly and distributed them uniformly on the surface of the substrate. Active and defective myosins were distinguished using different algorithm loops. Active myosin undergoes a nucleotide binding loop consisting of four main steps: binding to actin, performing a power stroke and propelling the actin filament, binding ATP and detaching from actin, and hydrolysing ATP to adenosine diphosphate and inorganic phosphate (ADP.Pi). Defective myosin undergoes one step: it binds to the actin filament, stays stuck, and detaches from the actin only if it experiences a force greater than the set detachment force. We adopted the threshold of the detachment force from the experiments, and in our simulation, a force beyond this threshold could cause either active or defective myosins to detach. To check the effect of defective myosin, we performed simulations with various active myosin ratios ranging from 10% to 100%. To validate the 3D simulation solution to the first problem, we developed a separate 1D mathematical model. In this mathematical model, we imitated the behaviour of defective motors by assuming that once bound to the actin filament, they stay bound and stretch along with the movement of the actin filament, and then snap off from the actin filament when the threshold of detachment force is exceeded. By analysing the results of the 3D simulation, we summarised the movement of the actin filament in the presence of defective myosin into three stages. When actin is stuck (active ratio < 90%), when actin wiggles with jerky movements (90% < active ratio < 92%), and when the actin filament is gliding continuously (active ratio >= 94%). In this simulation, the continuous glide stage starts when defective myosins are reduced to approximately 10%. This continuous gliding stage is the one required for micro-nano device applications. In the first two stages, the motility assay may not be useful because the translation of the actin filaments is negligible. This implies that, in an actomyosin-based biosensor, a high active myosin binding ratio is required to initiate continuous translation. This finding was validated by the 1D mathematical model, which showed that actin filament movement began when defective myosins were reduced to around 20%. The discrepancy in the percentages of the continuous stage between the 3D simulation and the mathematical model may be attributed to the exclusion of thermal fluctuations and other factors in the 1D mathematical model. Both the 3D simulation and the 1D mathematical model showed a narrow range within which the actin filament can glide continuously over myosin motors. Consequently, an engineer should carefully choose the substrate used for the actomyosin-based application to favour the adherence and retention of the active myosin motors. In the second problem, to improve the efficiency of cargo/drug delivery by actin-based molecular shuttles, we sought to contain the high flexibility of the actin filaments by applying an external force field. Similar to the 3D simulation in the first problem, we adopted the same simulation to model the second problem by adding an external force field. When an external force field was applied to the actin filaments, the high flexibility of the actin filaments was restrained, consequently improving the efficiency of transport. The results of this study and the modelling and simulation techniques are useful in highly targeted drug delivery technologies.

Index Terms — *Biomechanics, Modelling, Simulation, Biosensors, Biomechatronics, Biomedical Engineering.*

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF ALGORITHMS

ABBREVIATIONS

AF	Actin filament
ATP	Adenosine triphosphate
ADP	Adenosine diphosphate
MD	Motor density
Bio-MEMS	Biomedical microelectromechanical systems
HMM	Heavy meromyosin
LMM	Light meromyosin
🔖	Indexed word
$r_{substrate}$	Substrate active motor ratio
R	Active motor ratio $r_{substrate}$
L_p	Path persistence length

Chapter 1

INTRODUCTION

"Our main focus is a holistic understanding of the engineering principles applying to systems integrating molecular motors in general."

Accounts of Chemical Research
HENRY HESS *et al.*, 2018

This chapter introduces our study by providing important background information. Biomolecular motors are introduced and their engineering applications are discussed. Modelling and simulation are then introduced to outline their usefulness in research, particularly in this study. Finally, the problem statement is explained in detail.

1.1 Biomolecular Motors

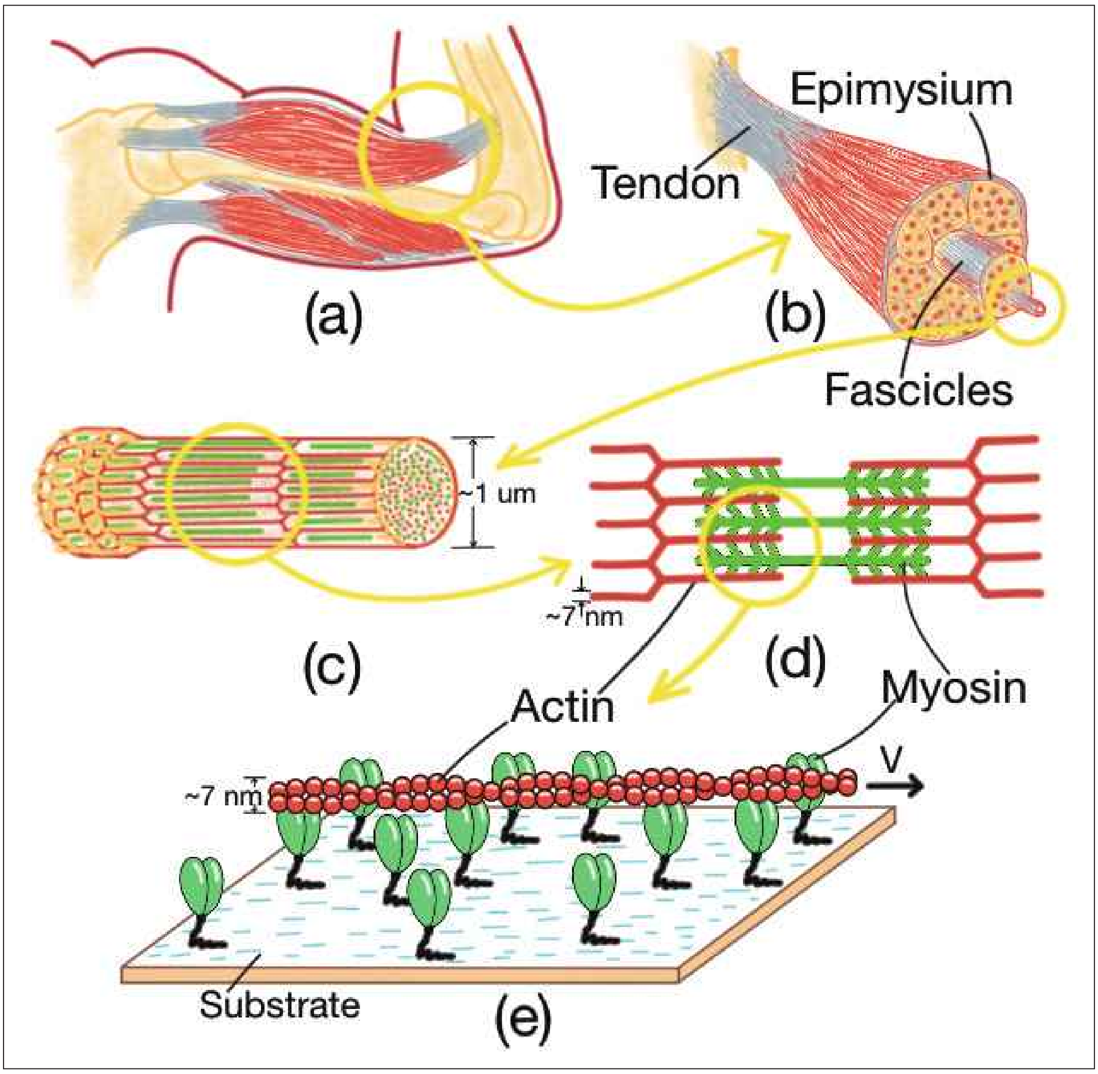


Figure 1.1: From a skeletal muscle to actomyosin-based device. (a) Skeletal muscle, (b) Parts of a muscle, (c) Myofibril, (d) The sarcomere: actin and myosin *in vivo*, (e) Actin and myosin in *in vitro* motility assay

1.1.1 The Muscle

Various parts of a living animal generate movement. To survive, animals feed on plants or other animals and reproduce. For this reason, most animals move to find food and mates. Food digestion and reproduction processes also involve various movements. These movements are the result of muscle contraction and relaxation. There are three types of muscles found in animals: skeletal muscle, cardiac muscle, and smooth muscle [1]. As seen in Figure 1.1(a), the skeletal muscles are attached to the bones and are responsible for the locomotion between the bones. Cardiac muscles are found in the heart and are useful in pumping blood throughout the body. Skeletal and cardiac muscles are both termed striated muscles, as they consist of repeated parallel streak patterns called sarcomere (see Figure 1.1(d)). Smooth muscles are found in vessels, veins, and hollow organs and are responsible for the involuntary movement of materials, through the contraction and relaxation of the actin and myosin filaments without sarcomere [1]. Despite the roles that muscle systems play in an animal, evolution has continued to favour various factors that promote muscle fitness, such as efficiency, speed, and low energy consumption [2].

Natural selection has optimised muscle construction according to animal requirements. A carnivorous animal, such as a cheetah, can run at a speed of around 104 km/h [3]. High speeds are required for predators and prey alike. However, this speed is occasional and is not necessary for herbivorous tortoises. Other animals, such as elephants, have optimised strength and ease of lifting. Today, the state of engineering motors has not yet matched the efficiency found in animal muscles. Understanding the natural solution to movement generation in animals may provide us with subtle solutions to achieving desired micro-scale movements.

Skeletal muscle consists of several fibres, which are essentially cylinder-shaped cells with a structure similar to that shown in Figure 1.1 (b). The myofibrils portrayed in Figure 1.1(c) are found in muscle cells and are approximately 1 μm in diameter, occupying more than 80% of muscle cells [4]. Around 2000 myofibrils are assembled to form a single muscle fibre [4], and additional assemblies make fascicles. The perimysium bundles up the fascicles, and the epimysium bundles up the perimysiums. Under microscopic observation, the myofibril shows distinct stripes called A and I bands, which are structures consisting of thick myosin filaments and thin actin filaments [5]. Figure 1.1 (d) shows the microscopic myofibil stripes that make up the sarcomere. The green stripes in Figure 1.1 (d) represent the myosin filaments, while the red stripes represent the Actin filaments. Figure 1.2 shows the same actin filament and myosin as in Figure 1.1 but in a little more detail. The thick myosin filaments are made up of myosin protein and are divided into two main parts: heavy meromyosin (HMM) and light meromyosin (LMM). HMM is divided into the binding part (S1), which is around 19 μm, and the hinge part (S2). LMM is the part that adheres to the substrate. The actin-thin filament is made up of tropomyosin and troponin. Actin filaments are rope-like with a diameter around 7 nm. It is astounding that these cytoskeletal proteins have the potential to build cytoskeletal structures of dimensions of up to 1000 μm.

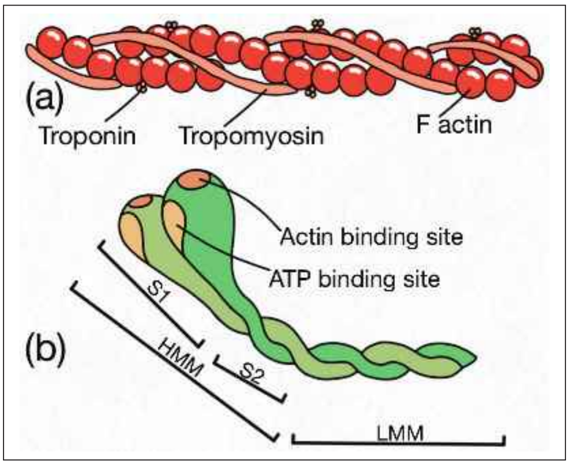


Figure 1.2: The actomyosin. (a) The actin thin filament structure, and (b) The myosin thick filament structure

1.1.2 *In Vitro* Motility Assay

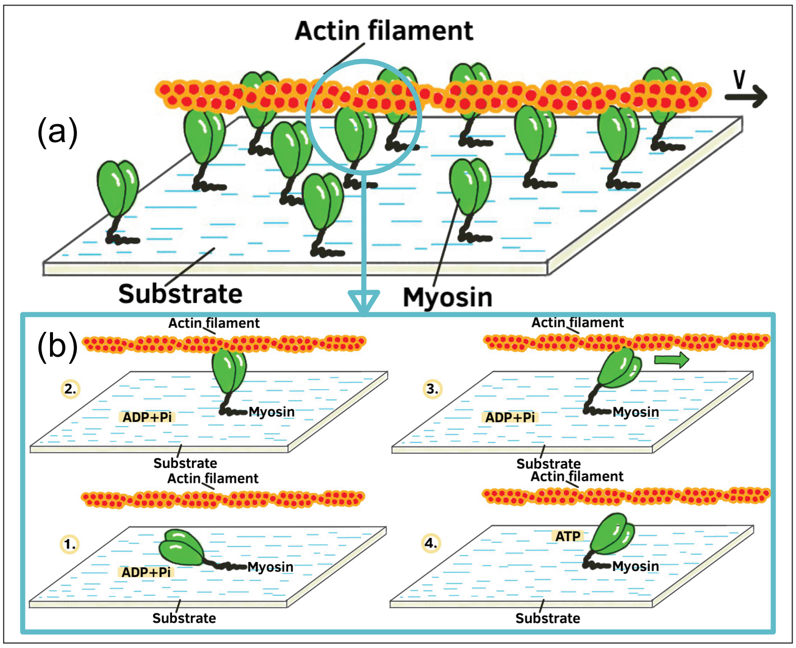


Figure 1.3: Myosin nucleotide binding. (a) *in vitro* motility assay, (b) myosin hydrolysis cycle

To understand the muscle mechanism or directly harness the functionality of biomolecular motors for artificial use, extraction from an animal muscle is performed. Figure 1.1 (e) presents an actomyosin motility assay *in vitro*. Since the first extraction of myosin on a glass surface *in vitro* motility assay by *Kron et al.* [6], various extraction protocols have been developed [7]. In general, actin and myosin are isolated from the muscle and then cleaned to minimise non-functional myosins. Various fluorescent techniques are used to help visualise actomyosin in action [8]. Figure 1.4 shows a *in vitro* motility assay in an experiment. An *in vitro* motility assay makes it easier to study various factors that affect or improve the actomyosin system. This kind of study is important to generate useful insights for *in vivo* and *in vitro*.

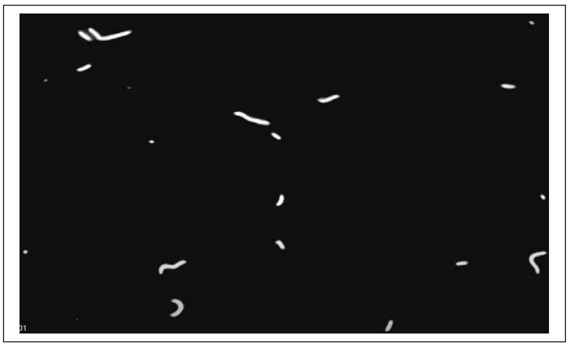


Figure 1.4: Actomyosin *in vitro* motility assay [9]. White lines are the actin filaments and the black background is the substrate.

For muscle contraction to occur *in vivo* and actin translation to occur *in vitro*, myosin undergoes four main processes. Figure 1.3 (b) shows the sliding contraction mechanism of the sarcomere, discovered by independent researchers [5]. In step ①, myosin hydrolyses ATP to ADP and inorganic phosphate, Pi (Equation (1.1)). This step is a recovery from the power stroke, and it allows myosin to relax, ready to bind to the actin filament. In step ②, myosin binds to the actin filament. In step ③, myosin contracts inward, which causes the actin filament to slide in. This step is called the power stroke, since the actin filament is propelled during this step. The chemical energy of ATP is converted directly to mechanical energy. In step ④, ATP binds to myosin, causing myosin to detach from the actin filament. Myosin can bind to the actin filament or ATP binds to myosin, but neither can happen at the same time. Sliding contraction occurs when steps ① to ④ are repeated, not necessarily in that order.

$$ATP + H_2O \longrightarrow ADP + P_i + Energy \tag{1.1}$$

1.1.3 Alternative Motors

The illustrations in Figure 1.1 focus on the basic parts that make up skeletal muscle and how actin and myosin can be extracted for applications *in vitro*. Although this figure may convey a systematic organisation and assembly of the muscle, the actual construction *in vivo* is rather complex [10]. In the cytoplasm of a cell, there exists a sophisticated network of interlinking proteins. This network is called the cytoskeleton. The cytoskeleton consists of protein filaments such as actin and the microtubule . The microtubules are tube-shaped with an outer diameter of approximately 25 *nm* and an inner diameter of approximately 18 *nm* [11]. Both microtubule and actin filaments grow out / polymerise from one end, called the plus-end. Cytoskeletal filaments combine to form large complex protein structures. For each filament, there is a corresponding motor protein that can bind to it. Motor proteins such as kinesin and myosin execute intracellular movements by progressively binding and detaching to microtubules or actin filaments, respectively. These movements play a key role in the organisation and shaping of cells. However, there are few microtubules in skeletal muscle [12], and actin and myosin have been shown to be the basic building blocks of muscle.

Biomolecular motors are molecular nanomachines that can achieve cold conversion of chemical energy into mechanical work. In an *in vitro* motility assay, they have several engineering applications. In contrast to purely artificial motors, biomolecular motors are thermodynamically stable. Furthermore, the efficiency of biomolecular motors and their low energy requirements make them suitable candidates for the production of low-energy synthetic micro- to nanoscale devices [13][14][15][16][17][18][19]. Using advanced manufacturing techniques such as microfabrication, biomedical microelectromechanical systems (Bio-MEMS) such as biosensors [20][21][22][23][24], biocomputers [25], molecular communication devices [26][27], self-assembling circuits [28] and artificial muscles for soft robotics [29].

Figure 1.5 shows various nanoscale motors that could be used as an alternative to make nanodevices instead of actomyosin. While kinesin transports material by walking towards the plus-end of the microtubule filament, dynein motors transport material by walking towards the opposite side, as shown by the black arrows in Figure 1.5 (a). Microtubule filaments can achieve speeds ranging from $0.2 - 0.4 \mu m/s$. F1-ATPase performs a rotary motion as shown by the arrow in Figure 1.5 (b) and is useful in metabolism. Figure 1.5 (c) shows one of the smallest man-made nanoscale motors that can act as an alternative to nanoscale actuation. Among these alternatives, biomolecular motors have a promising efficiency greater than $> 50\%$. Actin filaments can reach speeds that are ten times higher than those of microtubules.

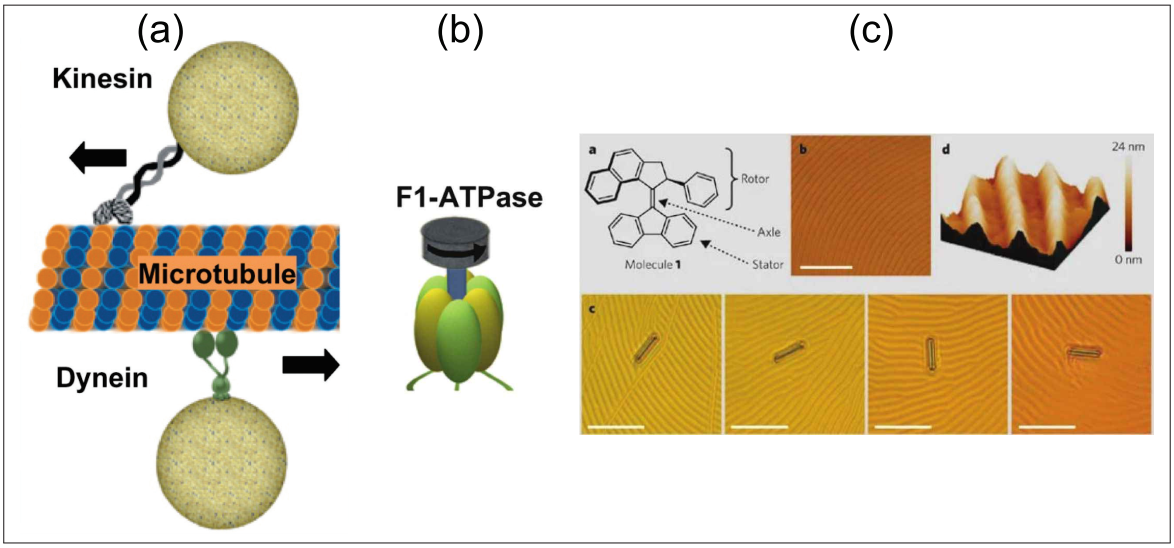


Figure 1.5: Alternative motors. (a) Kinesin and dynein biomolecular motor, (b) F1-ATPase rotational biomolecular motor [30], (c) Artificial nanoscale motor [31], [32]

1.2 Applications of Biomolecular Motors

1.2.1 Nanodevices Without Biomolecular Motors

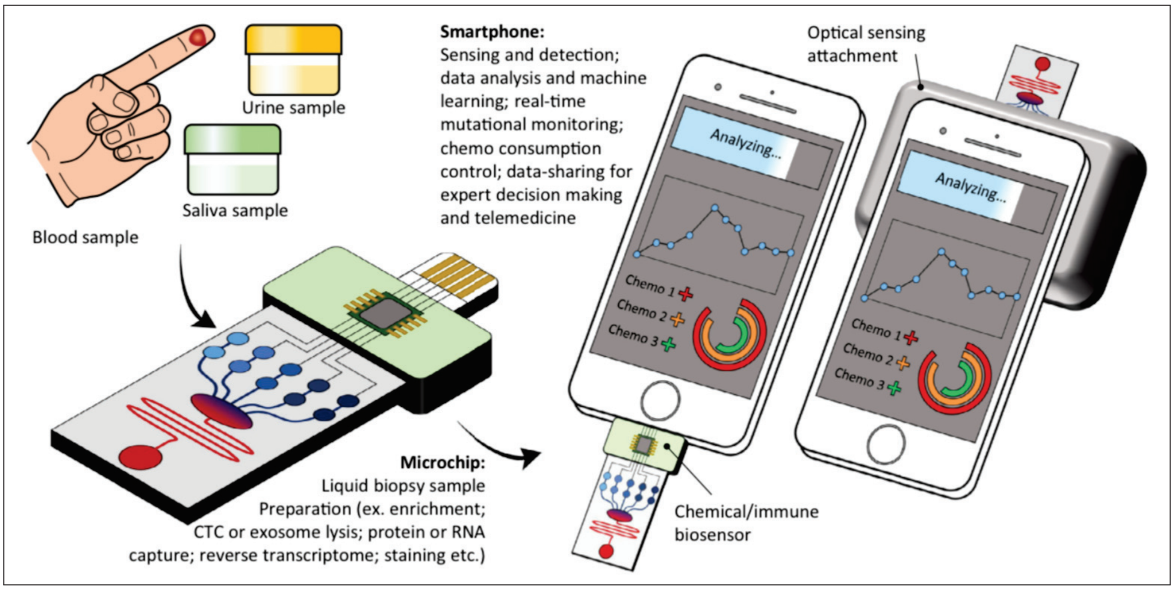

Urine sample
Saliva sample
Blood sample
Smartphone:
Sensing and detection; data analysis and machine learning; real-time mutational monitoring; chemo consumption control; data-sharing for expert decision making and telemedicine
Optical sensing attachment
Analyzing...
Chemo 1
Chemo 2
Chemo 3
Microchip:
Liquid biopsy sample Preparation (ex. enrichment; CTC or exosome lysis; protein or RNA capture; reverse transcriptome; staining etc.)
Chemical/immune biosensor

Figure 1.6: Biosensor application example. Pancreatic cancer management [33]

The ability to detect and understand small changes that occur around us has the potential to improve our lives. Given that our health is hugely dependent on the food we eat, being able to detect anomalies in the food we ingest is important. A case in point is the detection of milk allergens, which is helpful in combating complications that arise from milk consumption [34]. In medicine, the ability to detect disease quickly and easily has the power to save lives. A case in point is an inexpensive biosensor used for the early home detection of HIV/AIDS [35]. Figure 1.7 shows a paper-based lateral flow device, similar to the one used for early detection of HIV / AIDS. When the analyte sample is deposited, it moves along the device through capillary flow. As this process continues, target molecules are picked up and transported in various stages. If the sample in question contains suspected biomarkers, the test results will show a specific colour on the test line. Another example is the biopsies involved in cancer management, as seen in Figure 1.6. In addition, being able to administer the drug where it is required by the body can greatly reduce side effects, especially in cancer treatment [36], [37]. In forensics and road safety, biosensors, such as alcohol biosensors, have been helpful in identifying suspects and making it easier to restore order. Given that biosensors are low-energy devices, they hold the promise of providing various solutions to nanoscale problems.

In general, a biosensor is a device that is able to recognise the target biological or biochemical entity and convert that entity into a form that is easier to detect and analyse. For ease of analysis and storage, the output of most biosensors is an electrical signal. The quantities that biosensors seek to measure are usually in the nanoscale range, and thus call for the biosensor to be designed in the micro to nanoscale range. It is desirable that the biosensor does not destroy the biological systems it interacts with and does not compromise the signal being measured. Additionally, the results of the signal being measured must be reproducible for ease of analysis and verification.

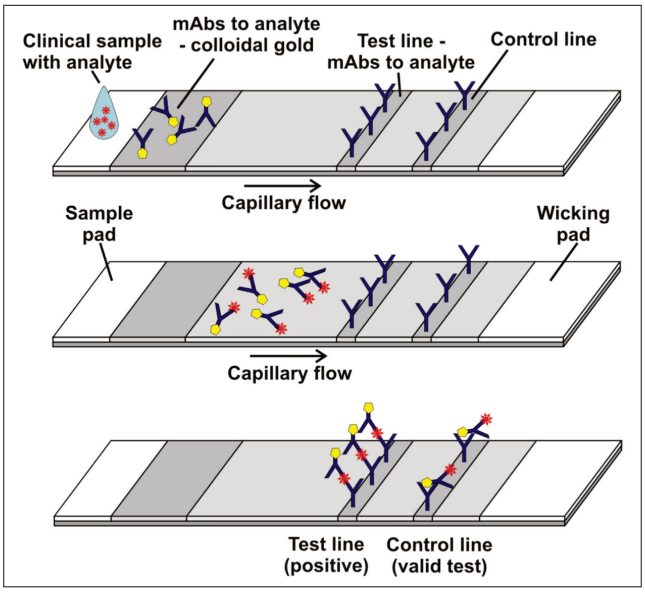

Clinical sample with analyte
mAbs to analyte - colloidal gold
Test line - mAbs to analyte
Control line
Capillary flow
Sample pad
Wicking pad
Capillary flow
Test line (positive)
Control line (valid test)

Figure 1.7: Biosensor application example. Early detection of HIV using lateral flow assay [38]

Electrochemical Biosensors

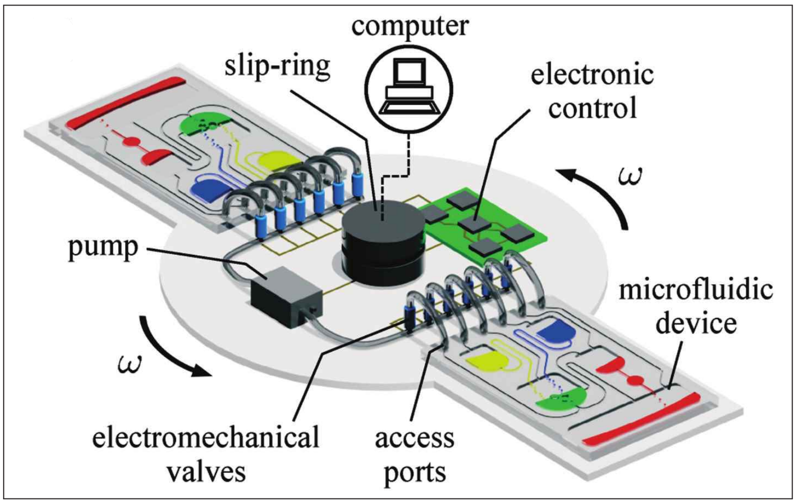


Figure 1.8: Lab on a chip [39]

Electrochemical biosensors are capable of providing analytical results directly by interacting with the biological entity. A good example is lab-on-a-chip (LOC) [40]. When the electrochemical potential is analysed, the interacting analyte can be identified and analysed. This may be done on the basis of current, voltage, resistance, or other electrical changes that occur during the interaction. This makes it possible to provide the analytical information directly and quickly. In a LOC device, various fluid tests can be conducted on a mm^2 space, making it possible to get ready results similar to conducting an experiment in a traditional laboratory. By programming the flow of fluids, fast and precise control can be achieved with reproducible results [39].

1.2.2 Biomolecular Motor-based Nanodevices

Biosensors

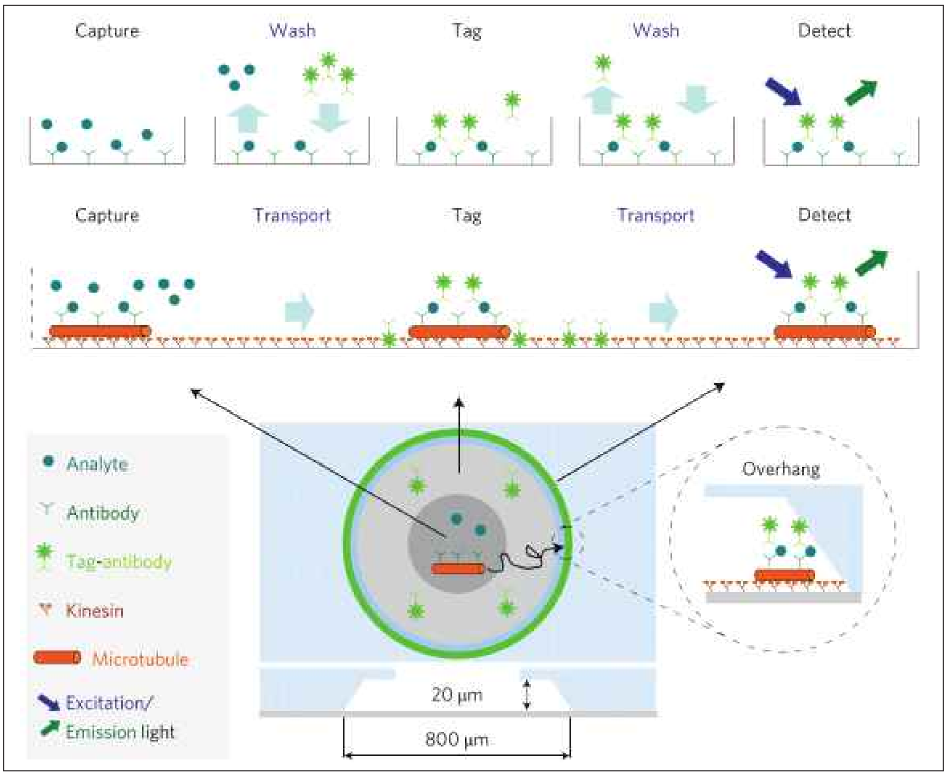


Figure 1.9: A design sequence of a biosensor driven by biomolecular motors [22]

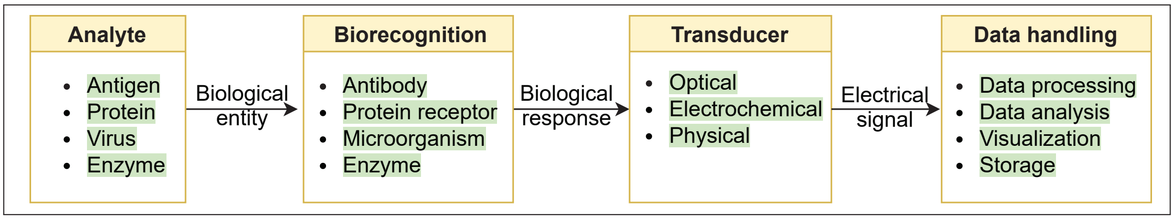


Figure 1.10: Biosensor components

Figure 1.9 illustrates the three main stages that a biosensor driven by biomolecular motors can undergo. In the first step, the cargo or analyte is loaded to the filament. In the second step, a fluorescent tag is placed on the cargo to aid in its detection. This set-up is also known as the molecular shuttle. In the last step, when the molecular shuttle delivers the analyte or cargo to the desired location, light is shown to the delivery location. By comparing the wavelength of the emitted light and that of the initial source, cargo delivery can be detected.

Figure 1.10 illustrates the four main components that make up a functional biosensor. The first component is the analyte, which is the entity to be detected and initiates the biosensor detection process. The detection or sensory component, also known as the bioreceptor, interacts with the analyte. Depending on the analyte in question, the result is unique. The signal transduction component converts the results from the sensory part to an electrical or optical signal. The signal processing component filters out any noise and amplifies the desired signal. The final part of the biosensor is the display or storage of the data obtained, depending on the intended application.

The basic working principle of a biosensor is the conversion of a biological entity to an electrical signal. For a biosensor to work, it needs to interact with the biological quantity being measured and provide the signal output of interest. There are various ways in which a biosensor can be made to detect an analyte. One way is by binding to the analyte, known as an affinity biosensor [41]. Another way is to react with the analyte, a method used by metabolic biosensors such as the flux sensor discussed by *Kochanowski et al.* [42]. Some biosensors combine with the analyte without a chemical reaction or active binding of the analyte. They are commonly known as catalytic biosensors. An example of a catalytic sensor is the catalytic hydrogen sensor [43]. The signal obtained from these kinds of detection is usually low and noisy and may need to be cleaned and amplified by microelectronics circuitry. Once the signal has been conditioned, it can be displayed or stored.

Molecular Recognition

Molecular recognition is achieved by forming a specific molecular bond with specific molecules. This may be done through static or dynamic methods. In a static method, two molecules interact to form a new compound, while in a dynamic method, the association constants are changed between the interacting molecules. In biological systems, molecular binding is common and is important in performing various processes in the body. These bonds include electrostatic, hydrophobic, hydrogen, and inter-abundance bonding [44]. Molecular recognition can also be achieved through specific geometric binding via a geometric imprinting process.

Optical Detection Methods

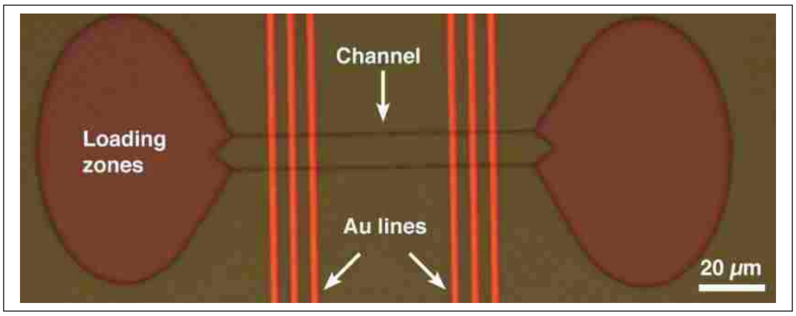


Figure 1.11: Actin fluorescence check [45]

In optical detection, the presence of the target molecule changes the properties of light such as phase shift, frequency, or amplitude. Figure 1.11 shows a novel fluorescence interference method to follow actin filaments [45]. In optical detection methods, a molecule can be detected either in its natural form or by using a fluoresce tag. This kind of detection is precise to one-molecule detection, and several parameters can be detected at the same time. In surface plasmon resonance, fluorescence labelling is not required. When the analyte binds, it causes a change in mass, and this change causes a change in the reflective index, which is detected as a change in resonance angle. This method is highly sensitive and interference-free. In the fluorescence tagging method, the analyte may be coupled to a reagent that responds to light intensity. When this light is shown and the analyte is available, a different light intensity is reflected back, which indicates the presence of the analyte. Other methods include fluorescence protein-based methods [44].

Noninvasive Biosensors

Noninvasive biosensors involve methods that do not penetrate the body to get samples to be analyzed. A good example is the saliva biosensor [46]. Saliva comes from the blood and carries metabolic information about the blood. Based on body condition, the analysis of the biomarkers found in saliva provides hints to a certain pre-existing condition. Similar results can be obtained from other metabolic by-products, such as sweat, breath, and tears. Their constituents may vary depending on the present condition of the body. However, noninvasive biosensors still face a challenge in the accuracy and reproducibility of the condition being tested.

Biocomputers

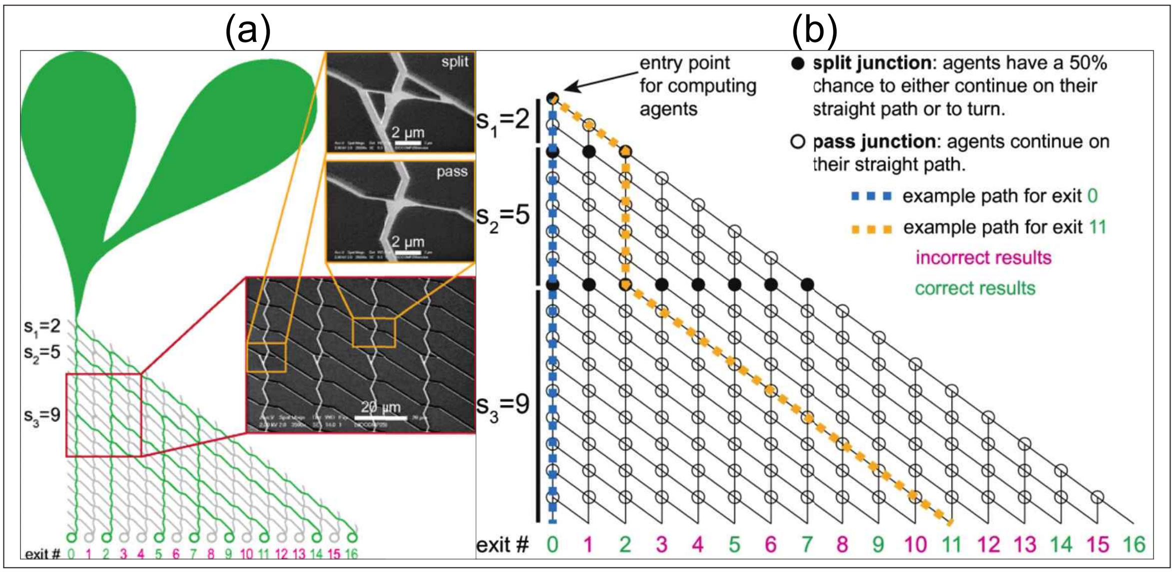

(a)
(b)
split
2 µm
pass
2 µm
20 µm
s_1=2
s_2=5
s_3=9
exit # 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
entry point for computing agents
● split junction: agents have a 50% chance to either continue on their straight path or to turn.
○ pass junction: agents continue on their straight path.
example path for exit 0
example path for exit 11
incorrect results
correct results

Figure 1.12: Biocomputation [25]

Biomolecular motors have the potential to implement low-energy molecular microcomputers that can solve combinatorial problems. Figure 1.12 shows the search for the complete NP solution using guided microtubule channels implemented by *Dan et al.* [25]. Silicon-based computers have done a great job in performing tasks sequentially. However, it is challenging for them to perform various tasks in parallel.

Microcomputers made from biomolecular motors could be engineered to solve these problems rather simply.

Actomyosin-based Biosensors

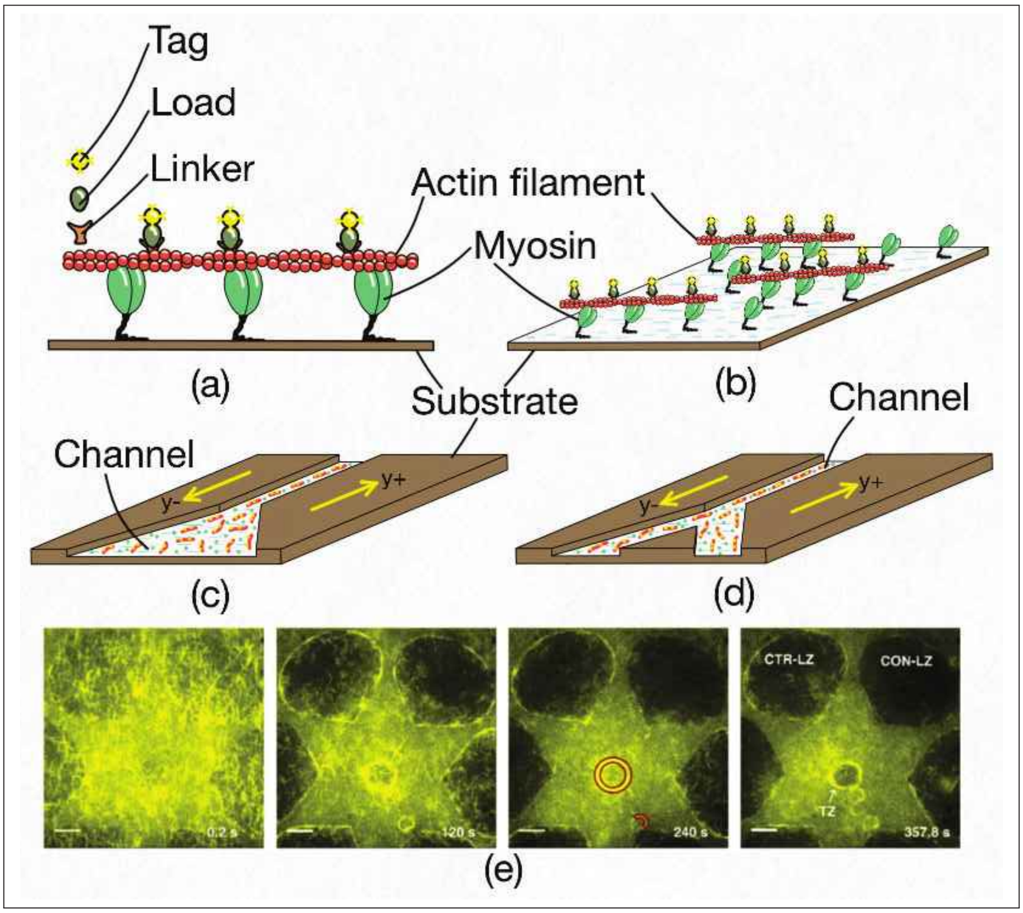


Figure 1.13: Components of biomolecular-based biosensors. (a) Molecular shuttle, (b) *In vitro* motility assay, (c) Guiding track used to concentrate (y+) or disperse(y-) load, (c) Guiding track used to divide(y-) or mix(y+) load, (e) Concentrator device [23]

A typical biomolecular-based biosensor usually consists of a molecular shuttle, a guiding track, and a transducer. As shown in Figure 1.13 , to make a molecular shuttle , linker proteins are attached to the actin filament and then the analyte or the cargo is attached to the linker. The cargo is then tagged with a fluorescent protein. This assemblage of tagged analyte linkers and the actin filament is driven by myosin motors adhered to the surface. Depending on the orientation of the adhered myosin, the actin filament and its cargo can move in a certain direction. A guiding track such as the one shown in Figure 1.13 (c) can be made to concentrate the cargo in a narrow channel or spread some cargo over a wide area. Figure 1.13 (d) shows another design of the guiding track that can divide the cargo into two channels or mix the cargo into one channel. At the analyte detection point or the cargo delivery point, the fluorescence excitation light is shown (Figure 1.13 (e)). If the cargo has reached the detection point, their fluorescent tags will emit light in response to the excitation light. Excitation and emitted light are optically differentiated and transduced into an electrical signal. Figure 1.13 (f) [23] shows an application of actin and myosin as a fast concentrator device.

Surface Roughness Detection

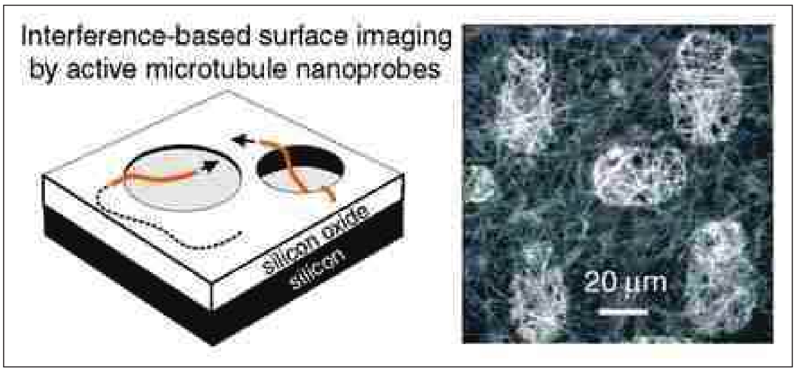

Interference-based surface imaging
by active microtubule nanoprobes
silicon oxide
silicon
20 µm

Figure 1.14: Surface imaging [47]

It is difficult to measure the roughness of a nanoscale surface. A biomolecular implementation such as that of actin or microtubules could be manipulated to assist in gauging nanoscale surface roughness. Figure 1.14 shows a tracking device implemented with microtubules, which can be used for nanoscale surface imaging [47]. To capture the surface roughness, biomolecular motors can be made to swipe around the surface, and through fluorescence imaging, the relative roughness of the surface being measured may be captured.

Concentrator Devices

Molecular concentrator devices may be used to converge or diverge the analyte. *Lard et al.* implemented a concentrator device made of actin and myosin as seen in Figure 1.10(f). This concentrator biosensor may be made portable and can achieve high speeds since it is integrated with actin and myosin. The self-driving nature of this device requires low energy to operate. This can be deployed for various tasks such as drug delivery.

1.3 Problem Statement

1.3.1 Molecular Shuttle Speed Limitation

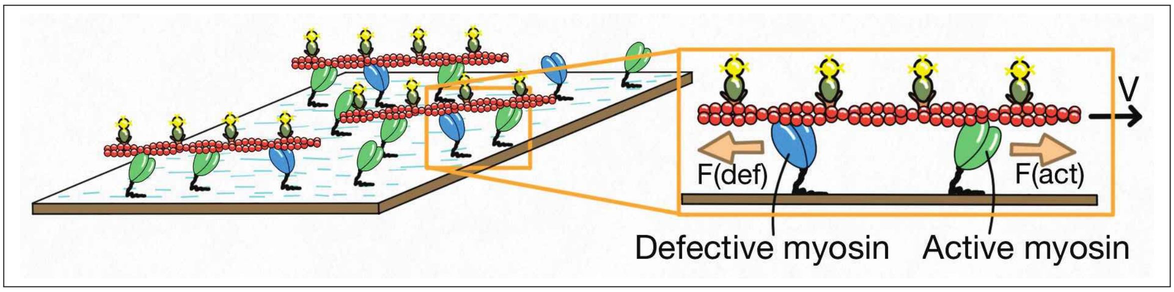


Figure 1.15: Molecular shuttles with defective myosins

A major problem in the implementation of molecular shuttles integrated with actin and myosin is the denaturation of myosin when it adheres to the surface of the substrate. Denatured myosin, hereafter referred to as defective myosin, prevents the transportation of molecular shuttle cargo and needs to be cleaned [48]. Figure 1.17 (a) shows a realistic actin and myosin motility assay. This motility assay consists of active (green) and defective (blue) myosins that interact with the actin filament (red). Figure 1.3 (a) shows an ideal assay condition consisting of only active myosin driving the molecular shuttle. Unfortunately, when the motility assay in an experiment, some myosins become defective when adhering to a surface as shown in Figure 1.17 (b). When the active motors complete the hydrolysis cycle, the defective motors get stuck on the actin filament. This means that both active and defective motors can bind to the actin filament but, unlike the active motors, defective motors do not hydrolyse ATP. They cling to the actin filament opposing the forward movement until a greater force dislodges them from the actin filament. Here, we seek to articulate how the presence of defective myosin in a motility assay affects the functionality of a biosensor integrated with actin and myosin motors.

1.3.2 High Flexibility of the Actin Filament

Although actin filaments have the advantage of attaining speeds higher than those of other filaments such as microtubules, their bottleneck has high flexibility (Figure 1.16 and [49]), making them exceptionally wiggly. The random movement of their trajectories contributed by the wiggly movement of the actin filaments' leading tips makes it difficult to predict the path that the designed molecular shuttles may follow. To design predictable devices, we need to understand how to control the random trajectory movements of the actin filaments.

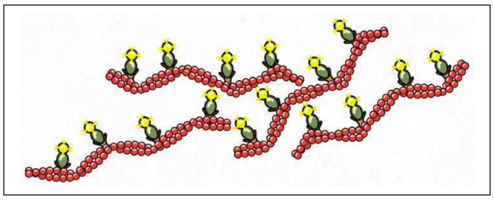


Figure 1.16: Actin filament high flexibility makes it to wiggle around uncontrollably.

The Problem Statement Summary

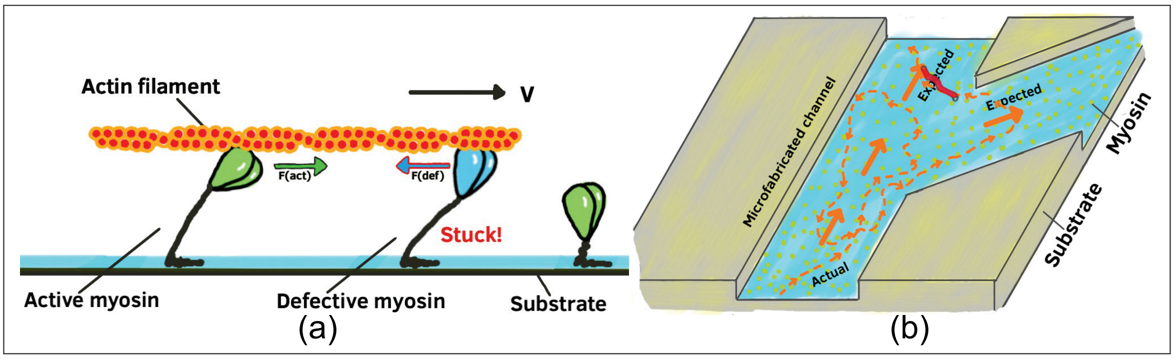


Figure 1.17: Problem statement summary: (a) Defective myosin gets stuck on the actin filament and resists the forward movement. (b) Actin filament high flexibility makes it to wiggle around uncontrollably.

1.4 Modelling and Simulation

Biomolecular motors are great for making microtransporters, such as molecular shuttles. Actomyosin has been shown to be superior in applications that require high speed. These biomolecular-based microtransporters have great applications in the development of biosensors. However, experiments alone can prove to be a daunting task in accelerating biosensor technology. Computational modelling and simulation come in handy in catalysing this discovery. In this study, we focus on the microtransporters powered by actin and myosin. We use simulation to articulate how the presence of defective myosin in a motility assay affects the functionality of a biosensor integrated with actin and myosin motors. In addition, we need to control the high flexibility of the actin filament with the aim of containing unpredictable movements. In a simulation, it is possible to precisely specify the active and defective myosin ratio. We can also set several parameters and perform simulations with high repeatability and low margin of error. Reproduction of the effects of the electric field on actin filaments as seen in the experiments will help us to understand and explore various design possibilities for actin-based biodevices. Although direct visualisation of filaments is currently advanced [50], as opposed to an experimental setup, a simulation can be performed with no spatio-temporal resolution limitations.

Figue 1.18 shows a summary of the simulation method. Most modelling and simulation techniques are implemented through a reductionist approach. The whole system is constructed by understanding and combining the knowledge of smaller subsystems of the whole system. These systems are turned into equivalent mathematical models which, in a simulation, are made to move through time in silico. However, some systems, such as nutrition, may not be well understood using the reductionist approach [51]. In such a case, simulating the whole system as a black box may still yield useful insights about the whole system. Modelling and simulation may be viewed as a tool that enhances our capacity to think. We can travel back in time by simulating historical phenomena, understand a current phenomenon deeper by using the latest information, or travel into the future by making plausible predictions. Systems such as BioMEMS implemented with biomolecular motors are difficult to understand using experiments alone. Computer simulation comes in handy in pushing scientific understanding even further.

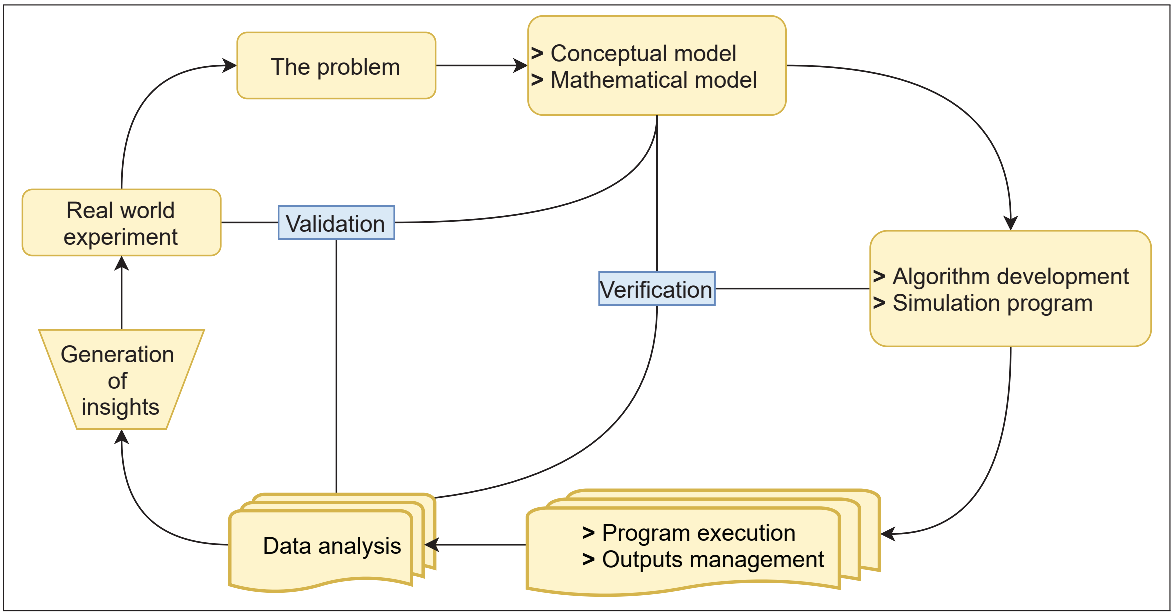


Figure 1.18: Modelling and simulation process

1.4.1 Modelling

The first step in modelling a biological system is to understand the underlying mechanism and principle of operation of the system being studied. Having identified the system to be studied, the operation principle of this system is broken down into modules or functions that represent a certain process or information about the system. The information used comes mainly from experiments performed in real life and the knowledge of mathematics and physics to make equivalent representations of the actual biological system. These representations can be solved analytically. This process is hereafter referred to as modelling. The modelling process involves the formulation of a concept and the making of appropriate mathematical formulations that represent or are equivalent to the concept. A validation process is required to check whether the constructed model actually represents the real-world problem being solved. These steps are repeated during the construction of the model to ensure that all known facts are included and well represented.

1.4.2 Simulation

In the simulation process (Figure 1.18), the models are executed in time with the help of a computer programme. The mathematical model representations are converted to computer programmes in a sequential scheme. When simulating subsystems that perform specific functions, the computer programme is developed in terms of modules that specify the algorithm the computer needs to follow. This process involves testing small portions of the developed programme and verifying that the expected results are obtained in agreement with the real-world problem.

Depending on the complexity of the programme, the simulation programme can be adjusted to optimise computer processing power and the time required for the simulation to finish. When the simulation programme is executed, it may involve inputting or outputting several files. These files need to be managed to ensure efficiency in programme execution. During the simulation process, programme development, programme execution, and input-output management are done repeatedly as needed.

1.4.3 Insights

The simulation results should be able to build evidence, suggestions, provide insight, or help with asking good questions about the main problem at hand. After the simulation program is completed, the output data is analyzed. The analysis results are compared with the expected results both from the real world experiments and with the expected analytical results from the mathematical models. If these information match, further analysis and simulation is carried out with the intent of gaining a deeper understanding of the system. If the simulation results and the expected results from the already performed simulation experiments do not match, further modelling and simulation execution is done to purge possible bugs. The iterative process of simulation, data anlysis, and comparison with the known facts about the system should point towards a deeper understanding, important predictions, or unearthing new knowledge about the overall system. It is important to validate every insight got from a simulation since it could be erroneous or a result of misinterpretation.

METHODOLOGY

> “A basic model has the advantage of a few experimentally accessible parameters that can directly be validated, and can focus on the design of interconnected structures.“
>
> *Lab on a Chip*
> TAKAHIRO NITTA *et al.*, 2006

In this chapter, the methods used in modelling and simulation are discussed. To begin with, the methods and steps that were followed during the development of the 3D simulation are discussed. The second section discusses the development of the 1D mathematical model and the last section explains how the external force simulation was developed.

2.1 Actin over Active and Defective Myosin: 3D Simulation

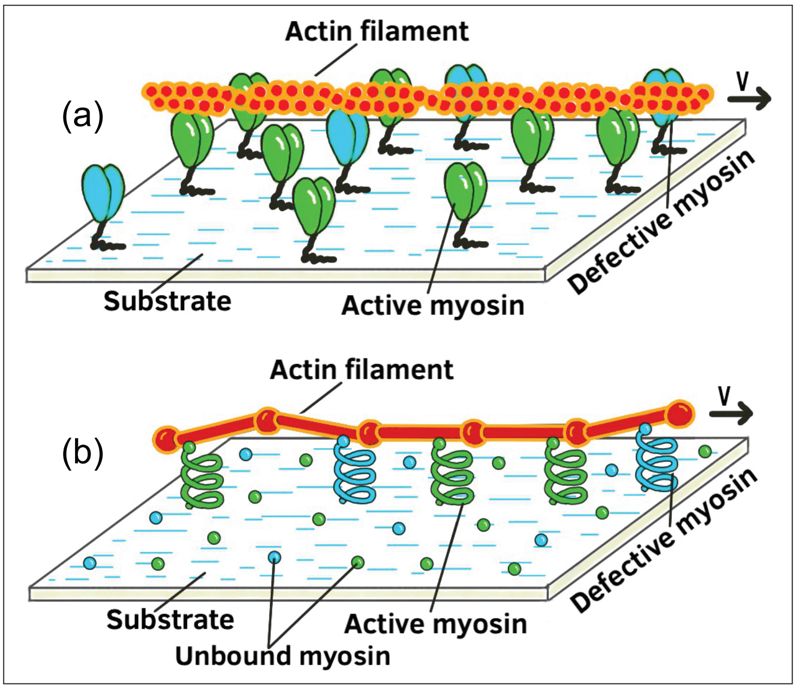


Figure 2.1: The simulation method. (a) the in vitro motility assay, (b) conceptual model of the in vitro motility assay - the in silico motility assay.

This section explains the mathematical equations that were employed in making the simulation program of the actin filament and myosin in an in vitro motility assay. An explanation of how the mathematical models may be executed in a computer program is provided. A pseudo code is also provided to elucidate a general implementation free from a specific computer language. In sections where it is applicable, references to the FORTRAN code that was used in this simulation are provided. The pseudo code in Algorithm 1 summarizes how various functions of the main code were implemented.

The simulation method used in this study is in part adapted and expounded from *Ishigure et al.* [52]. As illustrated in Figure 2.1, the actin filament is modelled as a set of rigid rods connected with beads. These rods can bend about the beads with a flexural rigidity, EI, set to 0.073 $pN.\mu m^2$ [49]. Both active and defective myosin are modelled as linear springs with spring constant of 300 $pN/\mu m$ [53], and detach force of 9.2 pN [54]. The difference between active and defective myosin is that while active myosin binds to the actin filament through the ATP hydrolysis cycle, the defective ones bind to the actin filament and only detach if forced to dislodge by the detach force. This actin-myosin simulation is implemented in a Brownian dynamics environment. The following are the key components that make up our simulation.

```
Declare variables and parameters;                                   // line 1-36
Set initial conditions;                                             // line 45-57
Prepare output files;                                               // line 60-76
Set initial conformation of the actin filament;                     // line 77-86
Set initial location and states of myosin tails;                    // line 87-91
function CALL CheckMotorFilamentProximity is                        // line 95-96
    Checking the proximity between the actin and myosin ;
function CALL InitialMotorBinding is                                // line 98-99
    Checking binding of motor at the initial time step;
function CALL CalculateForceMotor is
    Calculating the force acting on the motor;                      // line 101-103
    Calculating the force acting on the bead by motors;
    Calculating the total force acting on the bead;                 // line 104-110
Outputs;                                                            // line 112-128
repeat                                                              // line 130
    case Update motor states do
        function CALL MotorForcedDetachment is                      // line 135-139
            Forcing detachment of motors;
        function CALL MotorStateConv is                             // line 153-160
            Converting myosin states ;
        if Myosin binds to the actin then                           // line 140-152
            Check proximity between filament and motor;             // line 142
            Check if motors bind to the filament with binding rate $k\_a$ ;
            Check myosin state conversion upon binding;             // line 153-160
        function CALL RenewMotorPopulation is                       // line 161-167
            Renewing Motor Population;
    case Update bead positions do
        Generate random number matrices;                            // line 169-215
        Prepare vector representation for position, force, and random numbers ;
        repeat                                                      // line 216-240
            Perform unconstrained movements of beads
        until Constraint and confinement are Satisfied;
        Update position variables of beads;                         // line 249-259
    case Prepare force variables do
        function CALL CalculateForceMotor is                        // line 260-263
            Calculating the force acting on the motor;              // line 261-263
        function CALL CalculateForceBead is
            Calculating the force acting on the bead by motors;     // line 264
            Calculating the total force acting on bead;             // line 267-270
    Outputs;                                                        // line 271-291
until Set time elapses;
```

Algorithm 1: Actin-myosin motility assay main program. The line numbers referenced within the comments '//', refers to the main program in Appendix A.2

2.1.1 Unconstrained Actin Movement

To model the movement of the actin filament from one point to the next, we take into consideration the fluid drag, myosin tension, and thermal effects . In our simulation, the actin filament is made up of an assemblage of rigid rods joined together by bead points as illustrated by Figure 2.1. The movement of these bead points in this Brownian environment is unconstrained, and the global protein motion is assumed to be overdamped. The inertial forces are neglected since the viscous forces are much greater [11]. We used the explicit method by computing the later state of our system $X_i(t+\Delta t)$ using the current state $X_i(t)$ [55].

Figure 2.2 illustrates the derivation in Equations (2.2). In the following equations, let γ be the drag coefficient, BL be the actin filament bond length, $\overrightarrow{F_{m,i}}$ be the myosin tension force, $\overrightarrow{F_{a,i}}$ to be the actin filament bending, and $\overrightarrow{R_i}$ to represent the thermal fluctuations. The value 0.001 is the viscosity, η, of the surrounding fluid (water) and 0.006 is the actin filament diameter in μm [11], [56]. The variable $\sqrt{2D\Delta t}\overrightarrow{N}(0,1)$ is the diffusion coefficient contributed by thermal fluctuations and Brownian movement. The change in time, Δt, must be optimum in such a way that it is sufficiently small for better accuracy and not too small to make the computer computation time unnecessarily long.

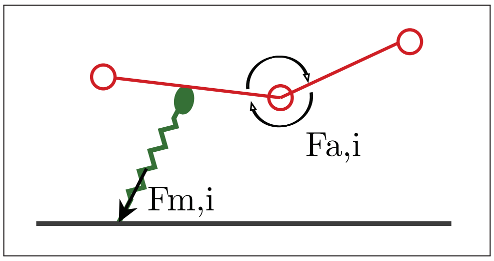


Figure 2.2: Unconstrained actin myosin interaction. Fm,i represent the force acting on myosin and Fa,i represent the bending of the actin filament

$$\gamma = \frac{3\pi \times 0.001 \times BL}{\log\{\frac{BL}{0.006}\}} \tag{2.1a}$$

$$D = \frac{k_B T}{\gamma} \tag{2.1b}$$

$$\gamma\frac{d\overrightarrow{X_i}}{dt} = \overrightarrow{F_{m,i}} + \overrightarrow{F_{a,i}} + \overrightarrow{R_i} \tag{2.2a}$$

$$\gamma\int_t^{t+\Delta t}\frac{d\overrightarrow{X_i}}{dt}dt = \int_t^{t+\Delta t}\overrightarrow{F_{m,i}}dt + \int_t^{t+\Delta t}\overrightarrow{F_{a,i}}dt + \int_t^{t+\Delta t}\overrightarrow{R_i}dt \tag{2.2b}$$

$$\int_t^{t+\Delta t}d\overrightarrow{X_i} = \frac{\Delta t}{\gamma}\overrightarrow{F_{m,i}} + \frac{\Delta t}{\gamma}\overrightarrow{F_{a,i}} + \frac{1}{\gamma}\int_t^{t+\Delta t}\overrightarrow{R_i}dt \tag{2.2c}$$

$$\overrightarrow{X_i}(t+\Delta t) - \overrightarrow{X_i}(t) = \frac{\Delta t}{\gamma}\overrightarrow{F_{m,i}} + \frac{\Delta t}{\gamma}\overrightarrow{F_{a,i}} + \sqrt{2D\Delta t}\,\overrightarrow{N}(0,1) \tag{2.2d}$$

$$\boxed{\overrightarrow{X_i}(t+\Delta t) = \overrightarrow{X_i}(t) + \frac{\Delta t}{\gamma}\overrightarrow{F_{m,i}} + \frac{\Delta t}{\gamma}\overrightarrow{F_{a,i}} + \sqrt{2D\Delta t}\,\overrightarrow{N}(0,1)} \tag{2.2e}$$

2.1.2 Actin Bending Force Calculation

The actin filament rigid rods are modelled in such a way that they can bend about beads. The amount of bending depends on the set flexural rigidity, EI, and the bond length of the rods that make up the actin filament. To elucidate the actin filament bending force, $\overrightarrow{F_{a,i}}$, the actin filament in Figure 2.3 is depicted as 3 beads ($\overrightarrow{R_1}$, $\overrightarrow{R_2}$, and $\overrightarrow{R_3}$) connected with 2 rigid rods. Here, $\overrightarrow{U_{12}}$ and $\overrightarrow{U_{23}}$ are unit vectors of $\overrightarrow{R_1}$-$\overrightarrow{R_2}$ and $\overrightarrow{R_2}$-$\overrightarrow{R_3}$, respectively. The bending force can be expressed as follows.

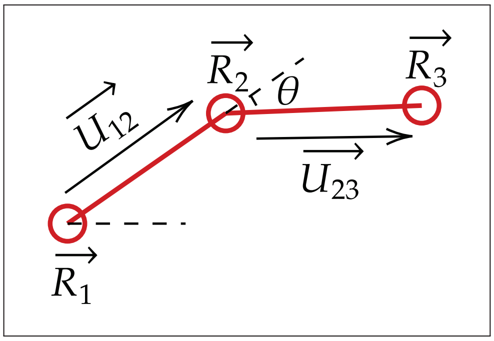


Figure 2.3: Actin filament bending

Assuming the number of beads = 3,

$$let\ U = \frac{1}{2}\frac{EI}{d^3}\sum_{i=2}^{n-1}(\overrightarrow{R_{i+1}} - 2\overrightarrow{R_i} + \overrightarrow{R_{i-1}})^2 \qquad [57](Eq.5) \tag{2.3a}$$

$$let\ F_i = \frac{1}{2}\frac{EI}{d^3}(\overrightarrow{R_3} - 2\overrightarrow{R_2} + \overrightarrow{R_1})^2 \tag{2.3b}$$

$$= \frac{1}{2}\frac{EI}{d}\left(\overrightarrow{U_{23}} - \overrightarrow{U_{12}}\right)^2 \tag{2.3c}$$

$$= \frac{1}{2}\frac{EI}{d}\left(\frac{\overrightarrow{R_3} - \overrightarrow{R_2}}{d} - \frac{\overrightarrow{R_2} - \overrightarrow{R_1}}{d}\right)^2 \tag{2.3d}$$

To calculate force from the potential energy, the acting bending force,

$$F_{a,i} = \frac{1}{2}\frac{EI}{d^3}\sum_{j=2}^{n-1}(\overrightarrow{R_{j+1}} - 2\overrightarrow{R_j} + \overrightarrow{R_{j-1}})^2 \tag{2.3e}$$

This can be implemented as shown in Appendix A.4

2.1.3 Actin and Myosin Contact

Figure 2.4 shows a segment of an actin filament (red line) and myosin (green spring) binding to it. This system is in a 3D space. Bead 1 represents a point on the actin filament segment located at a point $(X_{i,j}, Y_{i,j}, Z_{i,j})$ and bead 2 a point located at $(X_{i,j+1}, Y_{i,j+1}, Z_{i,j+1})$ towards the plus-end. The distance between bead 1 and the point of contact, c, is the intercept, e.

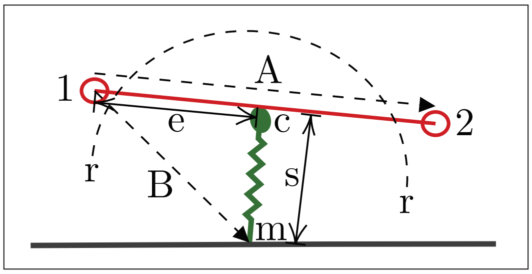


Figure 2.4: Actin and myosin contact

If the actin filament is within the radius, r, from myosin tail, m, the myosin can bind to it. This radius is hereafter referred to as the *capture radius* and it is taken to be 20 *nm* [58], [59], unless otherwise stated. The intercept e is given by the dot product of vector, $\overrightarrow{B}$, and vector, $\overrightarrow{A}$, divided by the actin bond length, B_L.

$$e = \frac{\overrightarrow{B} \cdot \overrightarrow{A}}{B_L} \tag{2.4a}$$

$$\overrightarrow{B} = \begin{bmatrix} Xm - X_{i,j} \\ Ym - Y_{i,j} \\ Zm - Z_{i,j} \end{bmatrix} ; \overrightarrow{A} = \begin{bmatrix} X_{i,j+1} - X_{i,j} \\ Y_{i,j+1} - Y_{i,j} \\ Z_{i,j+1} - Z_{i,j} \end{bmatrix} \tag{2.4b}$$

$$S^2 = |B|^2 + e^2 \tag{2.4c}$$

$$|B|^2 = \Delta x^2 + \Delta y^2 + \Delta z^2 \tag{2.4d}$$

$$\begin{bmatrix} Xc \\ Yc \\ Zc \end{bmatrix} = \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} + e - c \begin{bmatrix} X_2 - X_1 \\ Y_2 - Y_1 \\ Z_2 - Z_1 \end{bmatrix} \tag{2.4e}$$

From here, the possible contact points, c_{temp}, between bonds and myosin are calculated, and the myosin motor makes a stride. Based on the probability, the decision is made on whether the binding actually occurs and the unwanted points of no contact are discarded. Myosin binds to the actin filament via the shortest point from its tail,m, to the actin point of contact,c. Myosin step is made by taking *stepsize*/*bondlength*. The new myosin contact state is calculated as shown in Equation (2.5). A program to check for myosin-actin contact was implemented as shown in Algorithm 2.

$$ContactState = c_{temp} + \frac{Stepsize}{B_L} \tag{2.5}$$

```
SUBROUTINE CheckMotorFilamentProximity;                    // Appendix A.6
SUBROUTINE MotorBinding;                                   // Appendix A.7
SUBROUTINE MotorStep;                                      // Appendix A.8
SUBROUTINE MotorStuck;                                     // Appendix A.9
case SUBROUTINE CheckMotorFilamentProximity do
    if Active/defective motor is unbound, ready-to-bind ($M \cdot D \cdot P$ state), and within
    capture radius then
        Set TempContact;
        Leave ContactState the same;
case SUBROUTINE MotorBinding do
    if Active/defective motor is unbound, ready-to-bind ($M \cdot D \cdot P$ state), and within
    capture radius then
        case Active motors (SUBROUTINE MotorStep) do
            Make a step on actin filament;
        case Defective motors (SUBROUTINE MotorStuck) do
            Motor gets stuck on the filament;
    else
        Set TempContact = 0.0;
```

Algorithm 2: Filament proximity and myosin binding

2.1.4 Forces Acting on Actomyosin

In Figure 2.5, the green linear spring represents a myosin motor and the red rod represents the actin filament rigid rod. The actin filament has a unit length of 1 $(2-1=1)$. Force $F1$ acts on bead 1 while force $F2$ acts on bead 2 of the actin filament. Force Fm acts on the myosin motor. The delta, Δ, in Figure 2.5 represents the perceived center of mass.

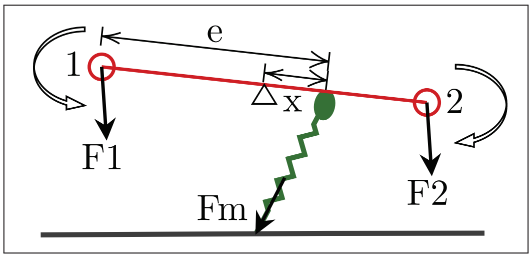


Figure 2.5: Forces actin on beads

The distance from the center of mass to the point where myosin binds to the actin filament is denoted by x. The intercept, e, is the distance between the first bead and the myosin binding contact on the actin filament. The myosin binding forces are derived in Equation (2.6). The sum of the counterclockwise moments, $\sum \curvearrowleft mom$, is equal to the sum of clockwise moments, $\sum \curvearrowright mom$.

$$\sum \curvearrowleft mom = \sum \curvearrowright mom \tag{2.6a}$$

$$\frac{1}{2} \times F_1 = (x \times F_m) + (\frac{1}{2} \times F_2) \tag{2.6b}$$

$$e = \frac{1}{2} + x;\ x = e - \frac{1}{2} \tag{2.6c}$$

$$\frac{1}{2}F_1 = (e - \frac{1}{2})F_m + \frac{1}{2}F_2 \tag{2.6d}$$

$$\boxed{F_m = F_1 + F_2} \tag{2.6e}$$

$$F_1 = F_m - F_2;\ F_2 = F_m - F_1 \tag{2.6f}$$

Substituting Equation (2.6f) into Equation (2.6d):

$$\frac{1}{2}F_1 = (e - \frac{1}{2})F_m + \frac{1}{2}(F_m - F_1) \tag{2.6g}$$

$$\boxed{F_1 = e \times F_m} \tag{2.6h}$$

$$\frac{1}{2}(F_m - F_2) = (e - \frac{1}{2})F_m + \frac{1}{2}F_2 \tag{2.6i}$$

$$\boxed{F_2 = (1 - e) \times F_m} \tag{2.6j}$$

2.1.5 Myosin Detachment

When an active or defective myosin is attached to the actin filament, its detachment from the filament depends on the force acting on it. The contact state of a given myosin is after its detachment has been handled. As illustrated in Algorithm 3, when the set detach force exceeds the actual force on the myosin head, the myosin is forced to detach from the filament and the force acting on the myosin is reset to 0. For the case of active myosin, the contact state is updated to -1 and for the case of defective myosin, the contact state is updated to 0.

```
CALL MotorForceDetachment;                              // Appendix A.11
Input: F_Motor_X, F_Motor_Y, F_Motor_Z, Release_ADP, ContactState
case Active motors do
    if force acting on motor heads > detachment force then
        Set the force acting on the motor to be 0;
        Set the ContactState to be $-1.0(M \cdot ATPstate)$;
        Set the $ReleaseADP = 0$;
case Defective motors do
    if force acting on motor heads > detachment force then
        Set the force acting on the motor to be 0;
        Set the contact state to be 0.0 (unbound state);
Output: F_Motor_X, F_Motor_Y, F_Motor_Z, Release_ADP, ContactState
```

Algorithm 3: Forced detachment of motors

2.1.6 Conversion of Myosin States

The myosin nucleotide binding used in our simulation followed the steps illustrated in Figure 2.6. An active myosin motor undergoes these four main steps in a hydrolysis cycle. Myosin step size was taken to be 10 *nm* [58].

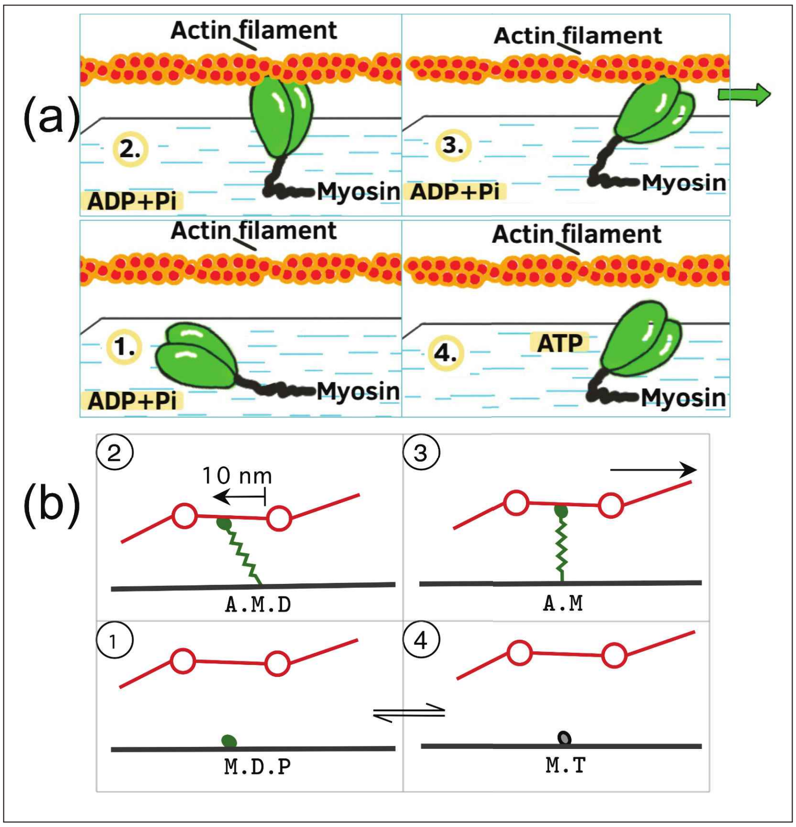


Figure 2.6: Myosin hydrolysis cycle. (a) myosin nucleotide binding states, (b) simulated conversion of active myosin states, adapted from *Ishigure et al.* [52]. A:Actin filament, M:Myosin, T:ATP, D.P: ADP+Pi, D:ADP

```
CALL MotorStateConv;                                   // Appendix A.15
Applies only to active motors;
case ① Myosin·ADP·Pi state (M.D.P: ContactState=0) do
    Convert to myosin·ATP state with the probability of $k\_hm * dt$;
case ② Actin·Myosin·ADP·Pi state (A.M.D: ContactState≥1; Release_ADP=0) do
    Calculate the tangential force;
    Calculate $k\_d$;
    Convert to Actin·Myosin state with the probability of $k\_d * dt$;
case ③ Actin · Myosin · ATP state (A.M: ContactState≥1; Release_ADP=1) do
    Convert to Myosin·ADP·Pi state with the probability of $k\_hp * dt$;
case ④ Myosin·ATPstate (M.T: ContactState=-1) do
    Convert to Myosin·ADP·Pi state with the probability of $k\_hp * dt$;
```

Algorithm 4: Motor state conversion

2.1.7 Myosin Placement

To make motility assays, we place myosin randomly but in a way that is uniformly distributed across the surface. We provide two sources of uniformly distributed random numbers, U_1 and U_2, of range $[0, 1]$, and then apply the Box-Muller transformation [60] to output independent random numbers with a standard normal distribution. The N value in Equation (2.7) is the output to be placed as myosin. This placement mimics the myosin motors on the surface of a glass plate in an in vitro motility assay.

$$N = \sqrt{-2\log U_1} \times cos(2\pi U_2) \tag{2.7}$$

Figure 2.7 illustrated how myosin motors were erased (E) and added (A) as the actin filament progressed throughout the simulation.

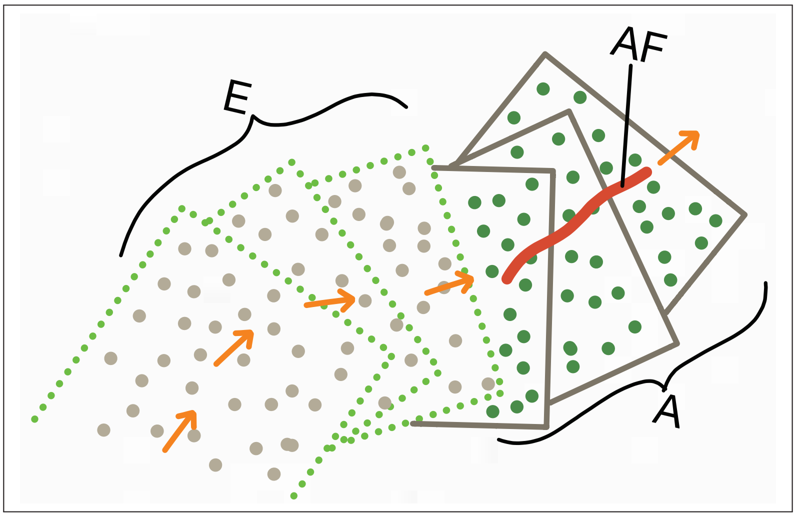


Figure 2.7: Myosin placement as the actin filament moves from one point to another.

```
SUBROUTINE RenewMotorPopulation;                          // Appendix A.14
Erase old motor region;
Update EraseCounter and ActiveMotorIdxOffset;
if Any bead is near boundary then
    Generate a new motor region;
    Generate temporal x- and y-coordinates of motor tails;
if Temporal x- and y-cordinates are appropriate then
    Update AddedMotorCounter;
    Accept the temporal coordinates as new XM and YM;
    Assign MotorType and ContactState;
```

Algorithm 5: Myosin placement

The Equation (2.8) was used in the simulation as the load dependence function.

$$k_d = k_d^0 \exp\left(-\frac{F\delta x}{k_B T}\right) \tag{2.8}$$

Table 1 shows a list of variables used during a typical simulation. Some of these values were changed accordingly as explained in the results and discussion section.

2.1.8 Simulation Validation

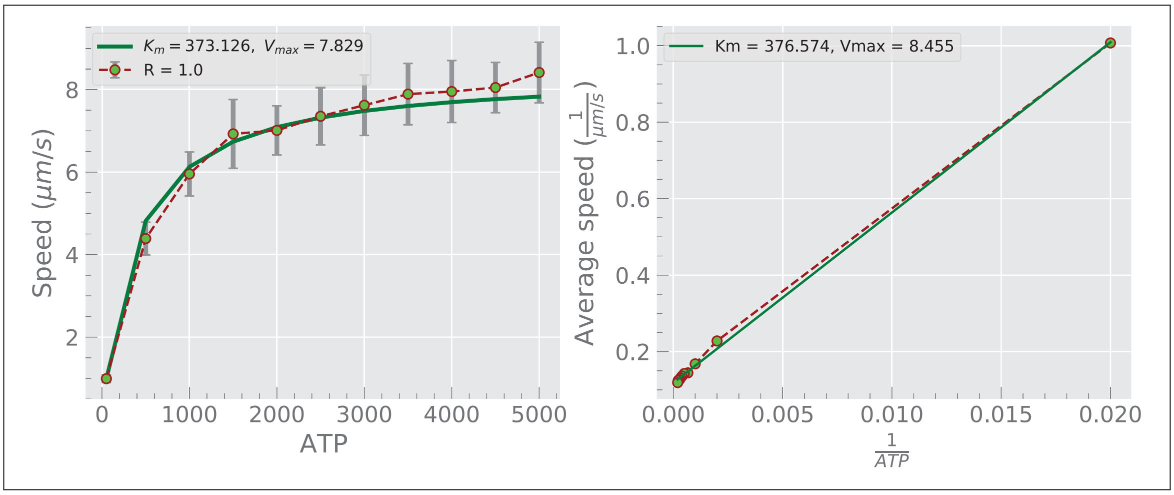


Figure 2.8: Michaeli Menten's fitting

$$y = mx + c \tag{2.9a}$$

$$\frac{1}{v} = \frac{K_m}{V_{max}} \times \frac{1}{ATP} + \frac{1}{V_{max}} \tag{2.9b}$$

$$V_{max} = \frac{1}{intercept}, K_m = V_{max} \times slope \tag{2.9c}$$

To validate that our simulation correlates with the Michaeli Menten's kinetics, we ran several simulations of varying ATP concentrations as shown in Figure 2.8.

Equation (2.9c) was used to calculate the maximum velocity and the K_m value.

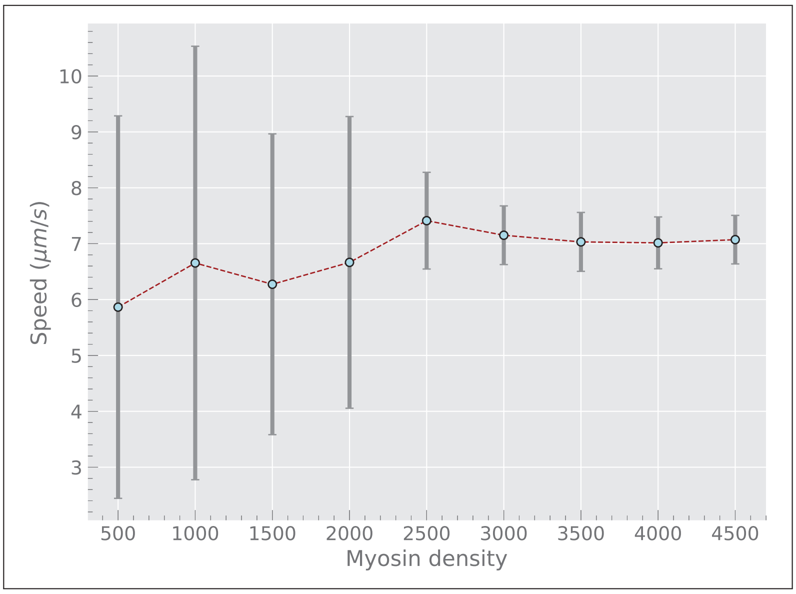


Figure 2.9: Speed changes with changing myosin density.

The actin filament speed stability with changing myosin density was tested in a simulated motility assay of 100% active myosin ratio. As plotted in Figure 2.9, it was observed that running the simulation at a low motor density of, say, 500 μm^{-2}, would yield highly fluctuating actin filament speeds characterized by detachment. For this reason, we ran all simulations at 3000 μm^{-2}, unless otherwise stated. A myosin density of 3000 μm^{-2} and above provides a relatively stable actin filament speed. However, a motility assay of around 4000 μm^{-2} would be unrealistic since it would be too cluttered with myosin.

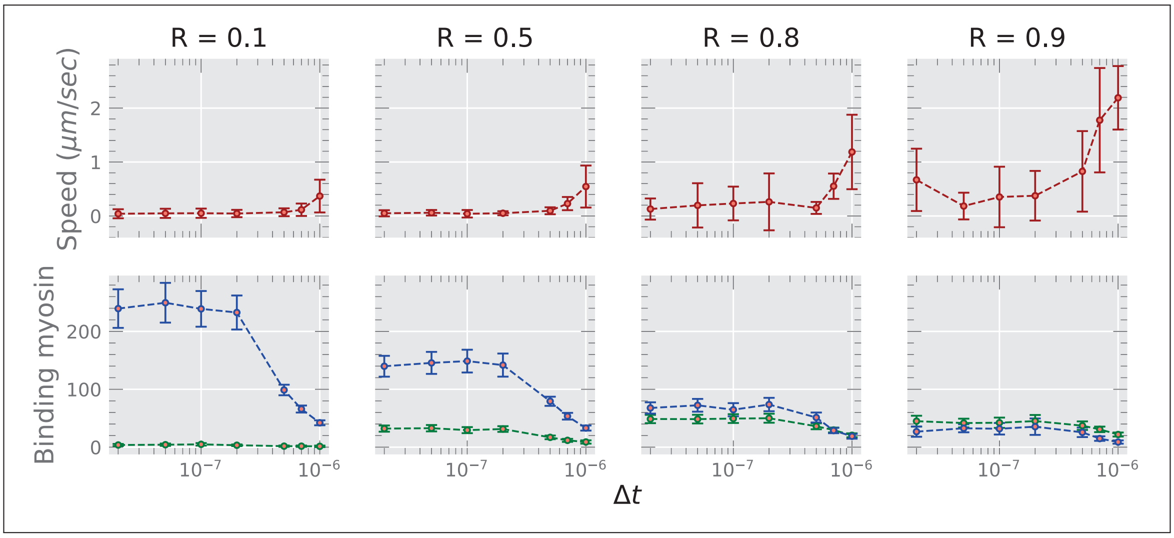


Figure 2.10: Variations caused by a change in Δt in simulation computations.

Before moving on to the simulation program, it is important to check if the preliminary results are stable and reliable. To carry out this test in our simulation, we ran several simulations of changing active ratios while changing the simulation Δt as shown in Figure 2.10. The smaller the precision of Δt, the more accurate our simulation computation is likely to be. However, if Δt is allowed to be too small, the computation time becomes unnecessarily long. In this simulation, we used $\Delta t = 1 \times 10^{-7}$ since at this point, the computations had already stabilized.

2.2 Actin Over Active and Defective Myosin: 1D Mathematical Model

This section explains the method used to develop a separate 1D mathematical model to validate the results of the 3D simulation. In this model, defective motors are represented in blue and are modelled to stay stuck to the actin filament once bound. As the actin filament is propelled by other motors, the bound defective motor is stretched to the point where it is forced to detach.

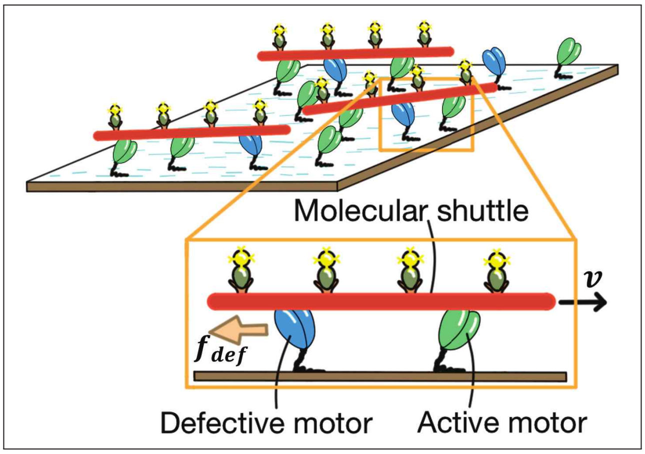


Figure 2.11: Active and defective myosin

The gliding speed, V, of the actin filament, AF, shown in Figure 2.11 is affected by a resisting force, f, caused by the defective motor. When f is zero, the actin filament will move at a maximum speed, V_{max}. However, as f increases, the actin filament slows down. As shown in Figure 2.12, if f is large enough ($= f_{stall}$), the actin filament will stop moving. Beyond f_{stall}, the actin filament may detach.

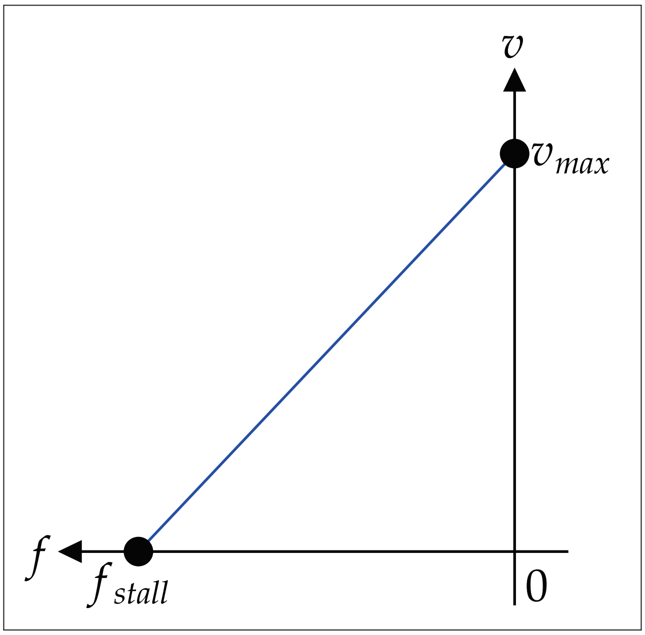


Figure 2.12: Active-defective force-velocity relationship: V_{max} vs. f_{stall}

The force-velocity relationship is expressed as in Equation (2.10), where V_{max} is the maximum gliding speed, f_{stall} is the stall force of a myosin, and f is the external force per active myosin. Note that $f, f_{stall} < 0$ because they are opposing the translation of the actin filament.

$$v(f) = \begin{cases} v_{max}(1 - \frac{f}{f_{stall}}), & \text{for } f_{stall} < f < 0 \\ 0, & \text{for } f < f_{stall} \end{cases} \tag{2.10}$$

Defective myosins effectively work as friction by impeding actin filament translation through repeated binding and dissociation. As illustrated in Figure 2.13, the defective myosin binds to the actin filament at a binding rate of $1/\tau_1$ and it stays stuck. As the actin filament continues to move, the defective myosin is made to elongate and eventually at a rupture force, f_{rupt}, after a time, τ_2, it is forced to detach from the filament. The time τ_2 depends on the actin filament speed and the rupture force of the defective myosin. Time T is the total time for the whole period from bind to detach. The time-averaged force generated by a defective motor, $\overline{f_{def}}$, as illustrated in Figure 2.13 and is derived as shown in Equation (2.11).

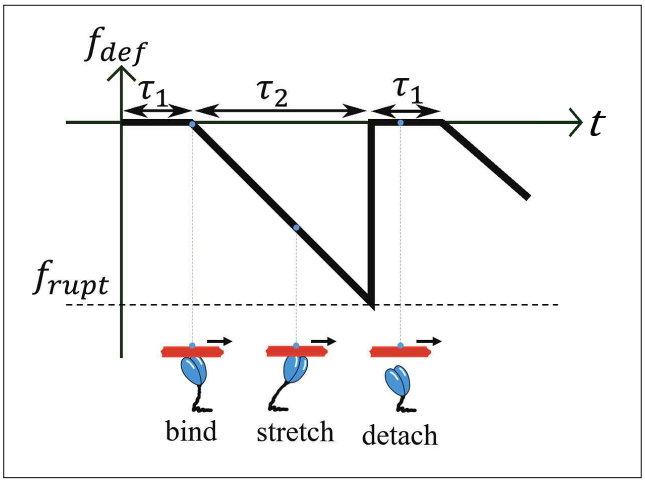


Figure 2.13: Binding and dissociation cycle of a defective myosin

$$\overline{f_{def}} = \lim_{\tau\to\infty} \frac{1}{T}\int_0^T f_{def}(t)\,dt \tag{2.11a}$$

$$= \frac{1}{\tau_1+\tau_2}\int_0^{\tau_1+\tau_2} f_{def}(t)\,dt \tag{2.11b}$$

$$= \frac{1}{\tau_1+\tau_2} \times \frac{1}{2} f_{rupt}\ \tau_2 \tag{2.11c}$$

$$= \frac{f_{rupt}}{2(1+\frac{\tau_1}{\tau_2})} \tag{2.11d}$$

$$\boxed{= \frac{f_{rupt}}{2(1-\frac{kV\tau_1}{f_{rupt}})}} \qquad \because\ \tau_2 = -\frac{f_{rupt}}{kv},\ \text{Hooke's law } f = ke \tag{2.11e}$$

Calculating the impedance per active motor, f_{imp}:

$$f_{imp} = \frac{\rho_d L}{\rho_a L} \times \overline{f_{def}} \tag{2.12a}$$

$$\boxed{= \frac{\rho_d}{\rho_a} \times \frac{f_{rupt}}{2(1-\frac{kv\tau_1}{f_{rupt}})}} \tag{2.12b}$$

Steady state gliding speed:

Substituting Equation (2.12b) into Equation (2.10):

$$v = v_{max}\left(1 - \frac{\rho_d}{\rho_a} \times \frac{1}{2(1 - \frac{kv\tau_1}{f_{rupt}})} \times \frac{f_{rupt}}{f_{stall}}\right) \tag{2.13a}$$

$$\left(1 - \frac{kv\tau_1}{f_{rupt}}\right)v = v_{max}\left((1 - \frac{kv\tau_1}{f_{rupt}}) - \frac{f_{rupt}}{2f_{stall}} \times \frac{\rho_d}{\rho_a}\right) \tag{2.13b}$$

$$v - \frac{k\tau_1}{f_{rupt}}v^2 = v_{max} - \frac{kv_{max}\tau_1}{f_{rupt}}v - \frac{v_{max}f_{rupt}}{2f_{stall}} \times \frac{\rho_d}{\rho_a} \tag{2.13c}$$

$$\frac{k\tau_1}{f_{rupt}}v^2 - \left(1 + \frac{kv_{max}\tau_1}{f_{rupt}}\right)v + v_{max}\left(1 - \frac{f_{rupt}}{2f_{stall}} \times \frac{\rho_d}{\rho_a}\right) = 0 \tag{2.13d}$$

$$\boxed{v = \frac{f_{rupt}}{2k\tau_1}\left(1 + \frac{kv_{max}\tau_1}{f_{rupt}} \pm \sqrt{\left(1 + \frac{kv_{max}\tau_1}{f_{rupt}}\right)^2 + \frac{4kv_{max}\tau_1}{f_{rupt}}\left(1 - \frac{f_{rupt}}{2f_{stall}} \times \frac{\rho_d}{\rho_a}\right)}\right)} \tag{2.13e}$$

At the transition:

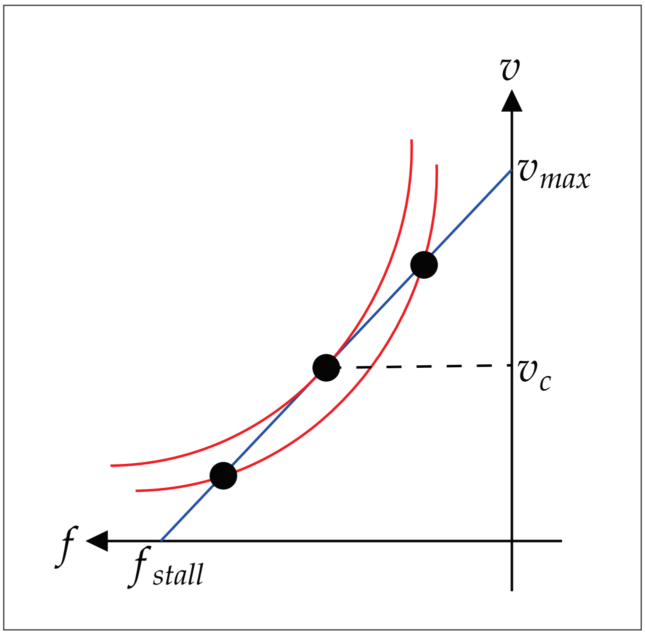


Figure 2.14: Transition point

$$v_c = \frac{f_{rupt}}{2k\tau_1}\left(1 + \frac{kv_{max}\tau_1}{f_{rupt}}\right) \tag{2.14a}$$

$$\boxed{= \frac{v_{max}}{2} + \frac{f_{rupt}}{2k\tau_1}} \tag{2.14b}$$

The transition is continuous if $v_c \leq 0$ and discontinuous if $v_c > 0$. The critical active motor ratio, R_c, for the discontinuous transition is derived as in Equation (2.15).

$$\left(1+\frac{kv_{max}\tau_1}{f_{rupt}}\right)^2-\frac{4kv_{max}\tau_1}{f_{rupt}}\left(1-\frac{f_{rupt}}{2f_{stall}}\times\left(\frac{\rho_d}{\rho_a}\right)_c\right)=0 \tag{2.15a}$$

$$\frac{4kv_{max}\tau_1}{f_{rupt}}\left(1-\frac{f_{rupt}}{2f_{stall}}\times\left(\frac{\rho_d}{\rho_a}\right)_c\right)=\left(1+\frac{kv_{max}\tau_1}{f_{rupt}}\right)^2 \tag{2.15b}$$

$$1-\frac{f_{rupt}}{2f_{stall}}\times\left(\frac{\rho_d}{\rho_a}\right)_c=\frac{f_{rupt}}{4kv_{max}\tau_1}\left(1+\frac{kv_{max}\tau_1}{f_{rupt}}\right)^2 \tag{2.15c}$$

$$\frac{f_{rupt}}{2f_{stall}}\times\left(\frac{\rho_d}{\rho_a}\right)_c=1-\frac{f_{rupt}}{4kv_{max}\tau_1}\times\left(1+\frac{kv_{max}\tau_1}{f_{rupt}}\right)^2 \tag{2.15d}$$

$$\boxed{\left(\frac{\rho_d}{\rho_a}\right)_c=\frac{2f_{stall}}{f_{rupt}}\left(1-\frac{f_{rupt}}{4kv_{max}\tau_1}\left(1+\frac{kv_{max}\tau_1}{f_{rupt}}\right)^2\right)} \tag{2.15e}$$

$$R_c=\left(\frac{\rho_a}{\rho_a+\rho_d}\right)_c \tag{2.15f}$$

$$\boxed{=\frac{1}{1+(\frac{\rho_d}{\rho_a})_c}} \tag{2.15g}$$

The critical active motor ratio, R_c, for the continuous transition is illustrated in Figure 2.15 and derived in Equation (2.16). Here, R represents the active motor ratio.

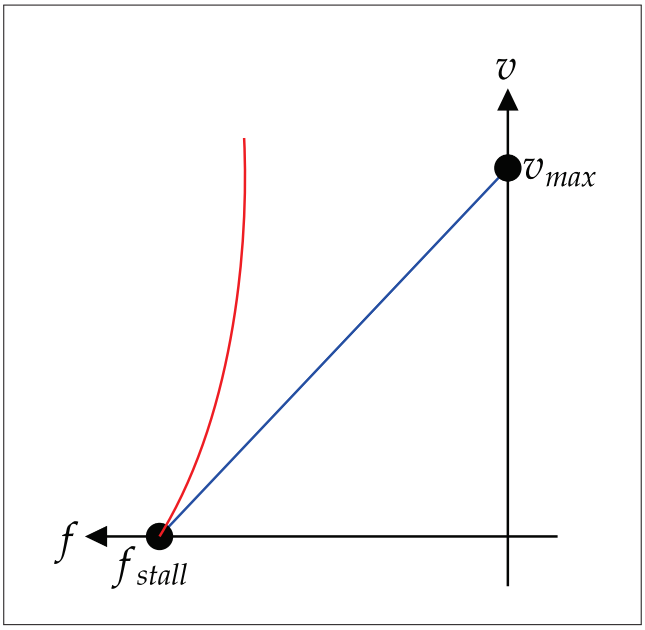


Figure 2.15: Continuous case

Injecting $v=0$ and $f_{imp}=f_{stall}$ into Equation (2.12b)

$$f_{stall} = \left(\frac{\rho_d}{\rho_a}\right)_c \times \frac{f_{rupt}}{2(1 - \frac{k \times 0 \times \tau_1}{f_{rupt}})} \tag{2.16a}$$

$$\left(\frac{\rho_d}{\rho_a}\right)_c = \frac{2f_{stall}}{f_{rupt}} \tag{2.16b}$$

$$\boxed{R_c = \frac{1}{1 + (\frac{\rho_d}{\rho_a})_c}} \because R = \frac{\rho_a/\rho_a}{\rho_a/\rho_a + \rho_d/\rho_a} \tag{2.16c}$$

2.3 Actin Under External Force: 2D Simulation

This section focuses on the method used to reproduce the effects of external forces on the actin filament as observed in the experiments [61] [62]. This was done by adapting the simulation method explained in Section 2.1 and incorporating the application of an external force. Figure 2.16 illustrates how the simulation was set up to exert an external force field on the actin filament as it moves in an in silico motility assay. Two kinds of external force fields were applied. First, the positive force field F(+), and secondly, the negative force field F(-). The positive force field was towards the movement direction of the actin filament v, while the negative force field was towards the opposite direction. The results of this simulation are as explained in Section 3.3.

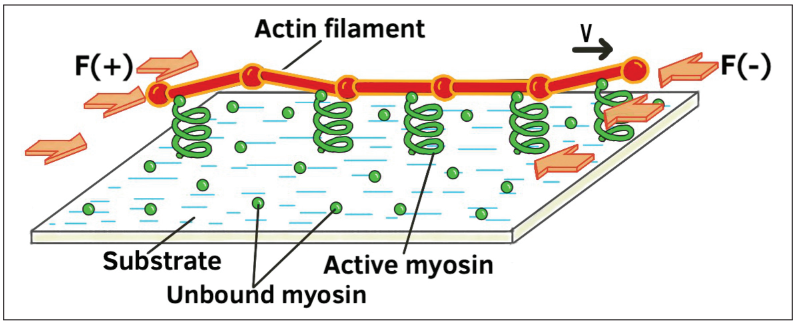


Figure 2.16: Applying external force field

2.4 Path Persistence Length Calculation

The path persistence length L_p, of a filament can be defined as the average length over which the trajectory of the filament remains straight. The persistence length of the actin filament was calculated as in Equation (2.17). In Figure 2.17 the dashed line represents the exact actin leading tip trajectory plotted from the simulation conformation output data points. The single lines (blue) joined with circles represent the approximate trajectory that can be easily predicted to predict where the actin filament is going. The approximate trajectory is the one used during the L_p calculation. The length of a single segment of the approximate actin filament trajectory (the length of line b_1, b_2, ...) can be varied by varying the Δt of the simulation data points to be used in the analysis. Taking a step size of 0 involves calculating the correlation of b_1 and itself b_1, that is, the dot product, $U_{b1} \cdot U_{b1}$. Taking a step size of 1 means calculating the correlation of b_1 with b_2, i.e. $U_{b1} \cdot U_{b2}$. Taking a step size of 2 means calculating the correlation of b_1 with b_3, i.e. $U_{b1} \cdot U_{b3}$, and so on. U_{b1} is a unit vector with the same direction as line b_1, and this continues up to the last line b_l and its unit vector U_{bl}. To check the straightness of a trajectory, say (x_1,y_1) to (x_3,y_3), we calculate the angle correlation between b_1 and b_2 by taking a dot product of the two unit vectors, $U_{b1} \cdot U_{b2}$. This is as shown in Equation (2.17c), and this correlation is equal to $cos(\theta)$. If the correlation is 1, the two lines are perfectly aligned to make a straight line. A value such as 0 would indicate no correlation and -1 would indicate an opposite perfect alignment. By comparing the various correlations formed by the actin filament leading tip, we can estimate the actin filament L_p.

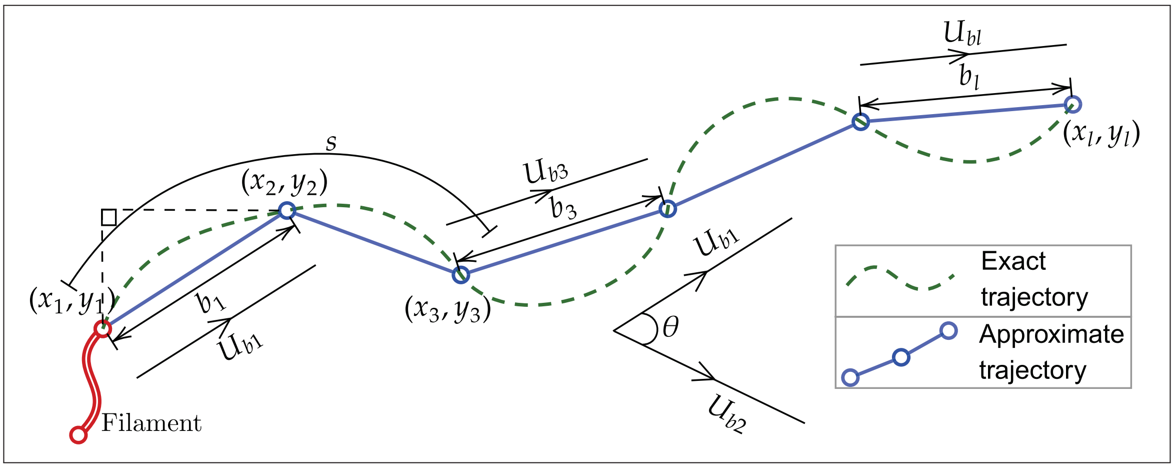


Figure 2.17: Actin persistence length

$$\|b_1\| = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{2.17a}$$

$$U_{b1} = \left(\frac{(x_2 - x_1)}{\|b_1\|}, \frac{(y_2 - y_1)}{\|b_1\|} \right) \tag{2.17b}$$

$$\boxed{cos(\theta) = U_{b1} \cdot U_{b2}} \tag{2.17c}$$

To calculate the average correlation, $\langle cos(\Delta\theta)(s)\rangle$, the Algorithm 6 is used. Δs is the step size taken. For example, if $\Delta s = 1$, the first correlation is calculated by taking the dot product, $U_{b1} \cdot U_{b2}$, and if $\Delta s = 2$, the first correlation is obtained by taking $U_{b1} \cdot U_{b3}$, and so on. Correlations in all steps are then calculated and then the average is evaluated in each case.

```
for $\Delta s = 0$ do
    Calculate $\langle cos(\Delta\theta)\rangle_0$;                    // step size = 0
    $= \text{mean}\langle U_{b1} \cdot U_{b(1+0)}, U_{b2} \cdot U_{b(2+0)}, U_{b3} \cdot U_{b(3+0)}, \ldots, U_{b(l-0)} \cdot U_{bl}\rangle$;
for $\Delta s = 1$ do
    Calculate $\langle cos(\Delta\theta)\rangle_1$;                    // step size = 1
    $= \text{mean}\langle U_{b1} \cdot U_{b(1+1)}, U_{b2} \cdot U_{b(2+1)}, U_{b3} \cdot U_{b(3+1)}, \ldots, U_{b(l-1)} \cdot U_{bl}\rangle$;
for $\Delta s = 2$ do
    Calculate $\langle cos(\Delta\theta)\rangle_2$;                    // step size = 2
    $= \text{mean}\langle U_{b1} \cdot U_{b(1+2)}, U_{b2} \cdot U_{b(2+2)}, U_{b3} \cdot U_{b(3+2)}, \ldots, U_{b(l-2)} \cdot U_{bl}\rangle$;
⋮
for $\Delta s = n$ do
    Calculate $\langle cos(\Delta\theta)\rangle_n$;                    // step size = n
    $= \text{mean}\langle U_{b1} \cdot U_{b(1+n)}, U_{b2} \cdot U_{b(2+n)}, U_{b3} \cdot U_{b(3+n)}, \ldots, U_{b(l-n)} \cdot U_{bl}\rangle$;
```

Algorithm 6: Actin filament trajectory average correlation. This algorithm is implemented in Appendix B.1(line 127-166)

Figure 2.18 shows the expected correlation plot. As the actin filament continues to move, the correlation of the leading tip decays. An implementation of Algorithm 6 and a L_p plot is shown in Appendix B.1.

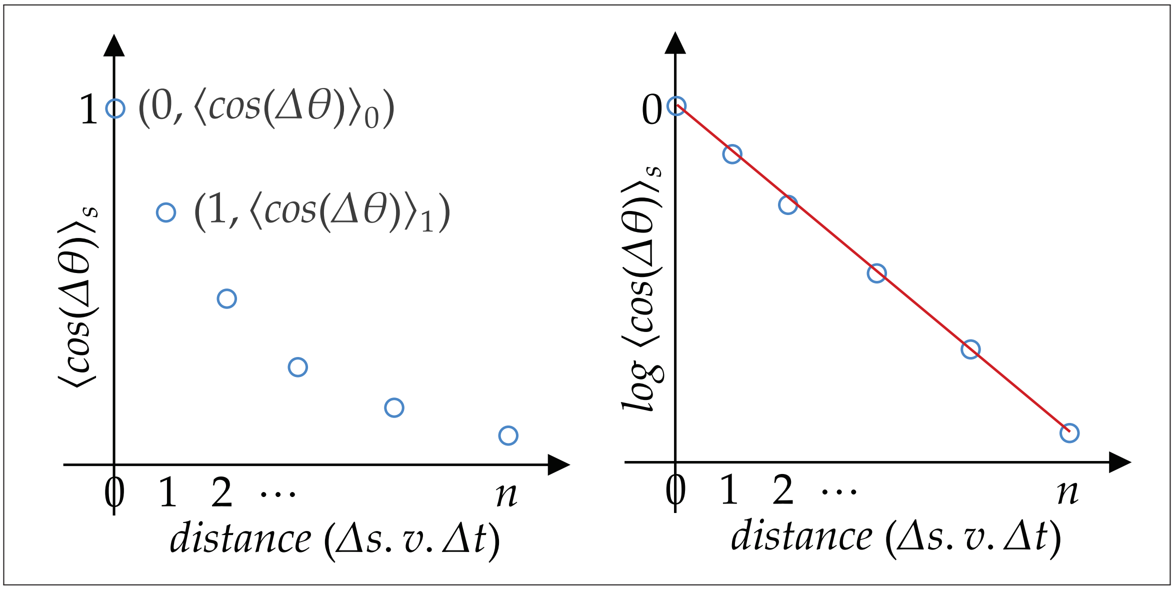


Figure 2.18: Persistence length example plot

$$S = \Delta s \cdot v \cdot \Delta t \tag{2.18a}$$

$$\langle cos(\Delta\theta)\rangle = \exp\left(-\frac{S}{2L_p}\right) \tag{2.18b}$$

$$\boxed{log\langle cos(\Delta\theta)\rangle = -\frac{1}{2L_p} \times S} \tag{2.18c}$$

As explained in Algorithm 6, $\langle cos(\Delta\theta)\rangle$ is the average correlation of the actin leading tip trajectory steps, S is the distance covered by the trajectory as in Equation (2.18a), total distance, $S_{total} = n \cdot v \cdot \Delta t$, and L_p is the path persistence legth of the trajectory.

As shown in Figure 2.18, L_p of the actin filament, L_p, can be calculated by calculating the gradient ($\frac{\Delta y}{\Delta x}$) of the line of best fit of Figure 2.18(right) and equating this gradient with $-\frac{1}{2L_p}$ of Equation (2.18c). Equation (2.18b) is derived from Jonathon Howard book [11] in pg. 111. When taking $log\langle cos(\Delta\theta)\rangle$, about 10% of the data may be used since as noticed in Algorithm 6, since a large step size is taken as n is approached, few data points are available towards the end of the trajectory thus losing statistical significance.

RESULTS AND DISCUSSION

> “The complexity of the protein-surface systems is greatly amplified by the polymer-fluid interface and the structure of molecular motors, making the study of these interactions critical to the success of molecular motor-based nanodevices.“
>
> *Biosensors and Bioelectronics*
> KRISTI L. HANSON *et al.*, 2017

In this chapter, the results for the three main projects are presented and discussed. These results include the 3D simulation of an actin filament propelled by myosin motors, the 1D mathematical model that describes the movement of the actin filament over defective myosin, and the movement of the actin filament under the influence of the external force field.

3.1 Actin over Active and Defective Myosin: 3D Simulation

In this section, the results of the actin-myosin 3D simulation are discussed in detail. The implications of these results are revealed along with the presentation of the results.

3.1.1 Actin Conformation and Trajectory

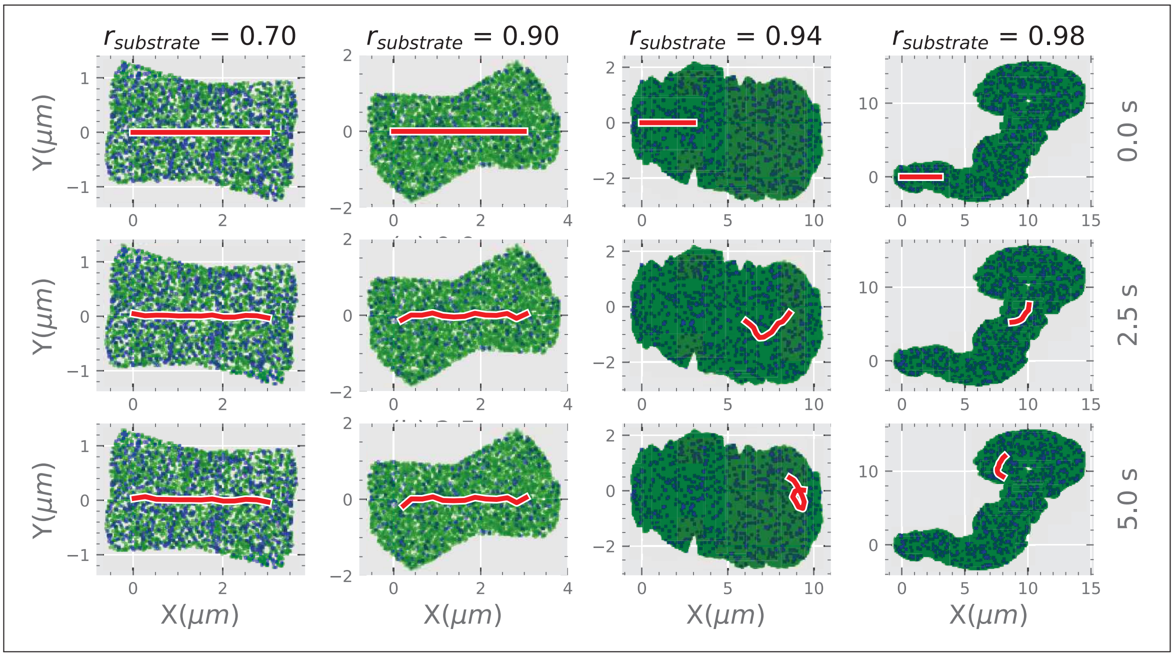


Figure 3.1: Actin filament conformation changes for various active motor ratios, $r_{substrate}$. $3\mu m$ actin filament, ATP = 2000 μM, and motor density = 3000 μm^{-2}. (a) $r_{substrate}$ = 0.70, (b) $r_{substrate}$ = 0.90, (c) $r_{substrate}$ = 0.94, (d) $r_{substrate}$ = 0.98. A video of this movement is available [63].

Actin Filament Conformation

Figure 3.1 shows the conformation of a $3\mu m$ actin filament that moves over active and defective myosins, as the active myosin ratio is systematically varied from $r_{substrate} = 0.1$ to $r_{substrate} = 1.0$. This movement of the actin filament undergoes three phases, hereafter referred to as the initiation phase (Figure 3.1(a)), the transition phase (Figure 3.1(b),(c)), and the gliding phase (Figure 3.1(d)). The initiation phase occurs at $r_{substrate} = 0.1$ to $r_{substrate} \leq 0.9$, where the actin filament remains stuck in approximately the same initial conformation throughout the simulation time. In the transition phase $0.9 < r_{substrate} \leq 0.92$, the actin filament wiggles around and occasionally makes jerky movements. These movements may eventually result in a progressive forward or curly movement. At $r_{substrate} > 0.92$, the actin filament enters the glide phase, which exhibits a continuous smooth procession. These changes are reminiscent of the microtubule movements discussed by *Scharrel et al.* [64].

By observing the position changes of the leading tip, we can have a bird's eye perspective of the whole actin filament translation. These changes in the trajectory of the filament can be used to estimate the relative distance travelled by actin as seen in Figure 3.2. In this study, the average speed of the actin filament and its L_p were calculated from the data from the trajectory of the tip. This comes in handy in predicting the performance of molecular shuttles in a biosensor application.

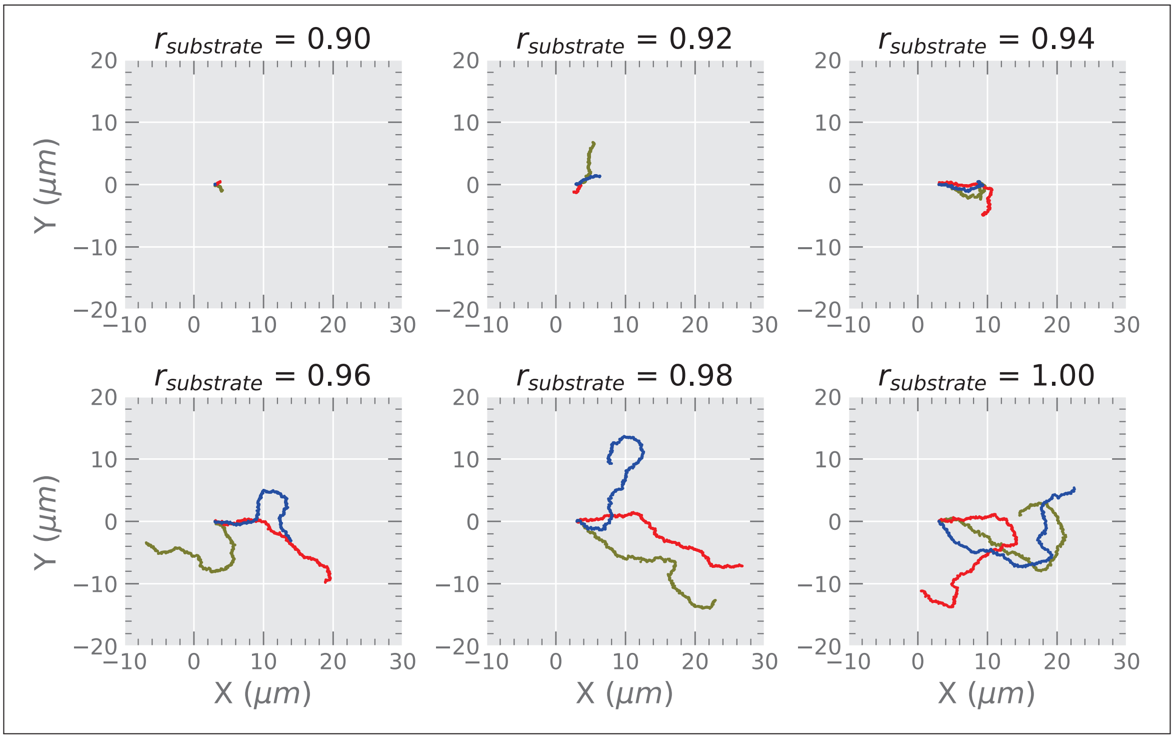


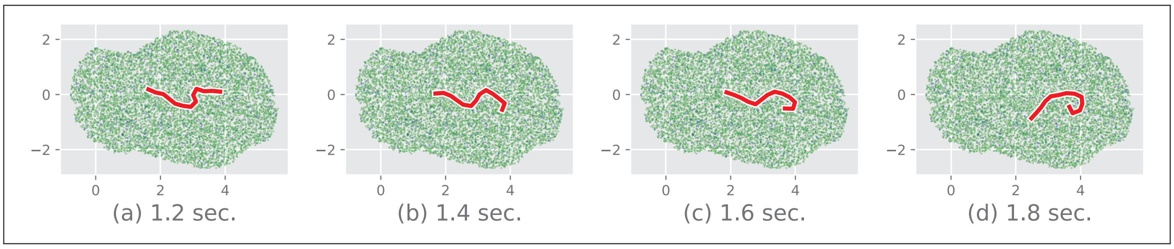
Figure 3.2: Actin filament trajectory

Actin Filament Initiation Phase

In the initiation phase, the relative movement of the actin filament is negligible. As seen in Figure 3.1(a) and Figure 3.2($r_{substrate}$ = 0.90), the actin filament is relatively straight throughout the simulation and the trajectory of the leading tip does not show progressive movement. Most defective myosins bind to the actin filament, since the active myosin ratio of the motility assay is the lowest at this stage. Although the middle part of the actin filament remains relatively straight with minimal fluctuations around the same position, the tip and tail segments exhibit more fluctuations. Although defective myosin binds throughout the length of the actin filament, the middle segments are coupled with other segments, damping down some fluctuations, whereas the leading and trailing tips are more exposed to the Brownian effects and thus fluctuating more.

Actin Filament Transition Phase

During the transition phase, the actin filament may conform to a variety of shapes, but the spiral shape is the most prevalent. Figure 3.3 shows typical screenshots of the movements of the actin filament during the transition phase. The leading tip of the actin filament may be captured, whereas the middle part and the trailing end continue to move. This portrays movements similar to the spiral defects described by *Bourdieu* *et al.* [65] and may serve to explain the mechanism of spiral defects and the resistance to smooth glide.



(a) 1.2 sec. (b) 1.4 sec. (c) 1.6 sec. (d) 1.8 sec.

Figure 3.3: Spiral defects. Transition phase movements at $r_{substrate} = 0.92$, $3\mu m$ actin filament, ATP = 2000 μM, and motor density = 2000 μm^{-2}

The mechanism of spiral movement of the actin filament during the transition phase may be attributed to its inherent structure. The structure of the actin filament is analogous to that of a mechanical chain. The flexural rigidity of the actin filament is such that it is easier to pull the entire filament than to push it. This asymmetry of forces creates a push-the-tail bias, a phenomenon that makes it easier to slide the actin tail and detach a defective myosin attached close to the trailing end than the one attached close to the leading tip. This promotes buckling, and hence curling, starting from the leading segment. This kind of asymmetry of forces has also been observed by *Koenderink et al.* [66], [67]. Figure 3.4 shows the simulation instances captured during the transition phase. If a point in the leading segment of the actin filament is caught by defective myosin, only that segment stays stuck, and its movement is inhibited. When the leading segment is stuck, the middle and tail segments continue to move. This is because it takes more time for the current detachment force to build up to the set critical detachment force (9.2 pN) that can remove the attached defective myosin. Before this force adds up, time elapses, and more defective myosins are still likely to bind to the stuck segment, holding it immobile for even longer. The transition phase trajectories seen in Figure 3.2 ($r_{substrate} = 0.92$) are short due to defective myosins that still inhibit progressive glide, thus the short distance travelled. Inhibition of the relative movement of the actin filament is similar to inhibition of the movement of the actin filament by Calponin discussed by *Shirinsky et al.* [68]. Every defective myosin that binds to the actin filament segment slows that segment and, eventually, the entire movement of the actin filament.

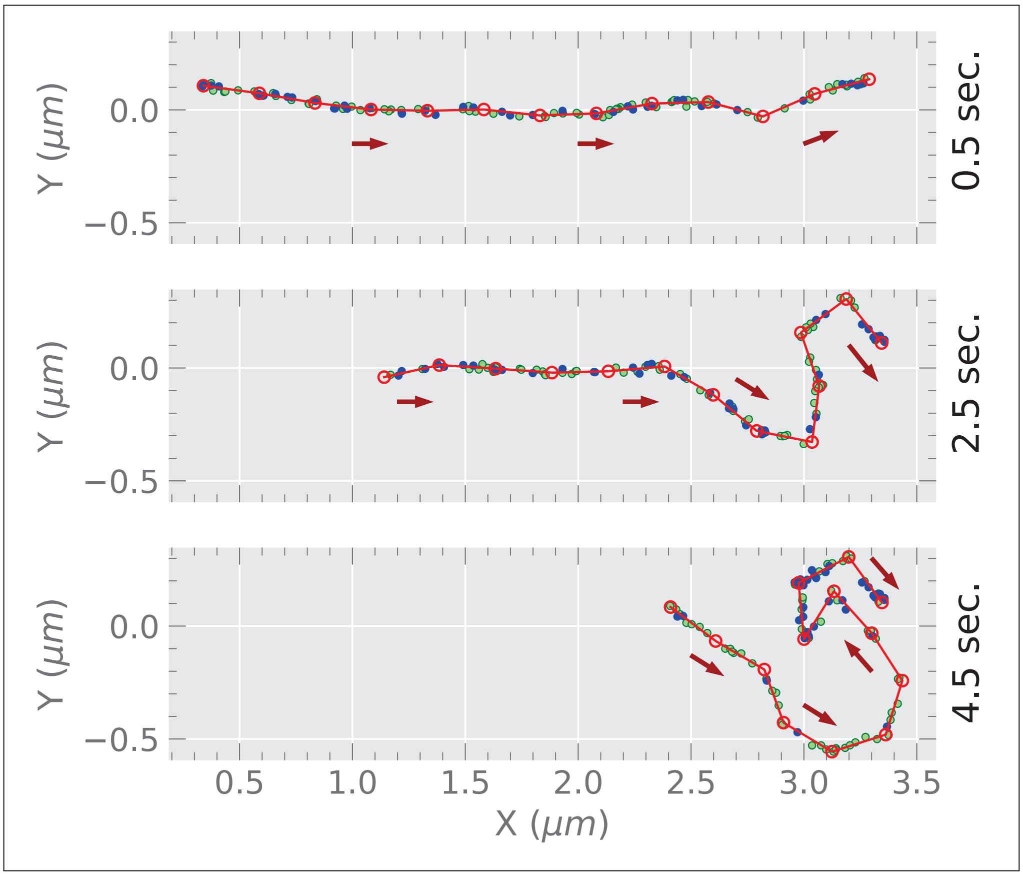


Figure 3.4: Push-pull force asymmetry at $r_{substrate} = 0.90$. Red line with red circles: the actin filament. Green dots: binding active myosins, Blue dots: binding defective myosin

Actin Filament Smooth Gliding

The movement of the actin filament on a motility assay rich in active myosin constitutes a smooth glide. Figure 3.1(d) and Figure 3.2($r_{substrate} \geq 0.96$) shows the smooth glide phase where the actin filament covers the longest distance. The distance travelled by the actin filament increases drastically with the high active myosin ratio of the motility assay. This shows that purification of the actin-myosin motility assay is important in reducing surface-adsorbed myosins that can inhibit movement. Although the effect of this process on actin speed may be unpredictable [48], affinity purification has been shown to improve motility [69]. To achieve smooth sliding of the actin filament, a sufficient composition of active myosin is essential in the motility assay. In our simulation, a composition of $\geq 10\%$ defective myosin was found to compromise motility and stop movement of the actin filament. Because defective myosins do not hydrolyse ATP, they bind permanently to the actin filament until they are forced to detach with greater force. Active myosins hydrolyse ATP and can collectively produce the impetus to cause bound defective myosins to detach from the actin filament. Reducing the defective myosin population to $< 10\%$ causes the glide of the actin filament to begin.

Actin Filament Path Persistence Length

The persistence length of a filament trajectory is the average length at which the trajectory remains straight. Knowledge of persistence length is helpful for gauging the ability to control the filament when designing biosensor guiding tracks. To calculate the persistence length of the actin filament trajectories, we adopted previously published methods [52], [70], and [11], as illustrated in Figure 2.17 in Section 2.4, in the methodology chapter.

3.1.2 Actin Filament Speed

Since the movement of the actin filament is stochastic, to calculate the velocity that captures the general direction of the actin filament, we need to choose an appropriate change in time Δt. In Figure 3.5, when choosing various Δt, the trajectory of the actin tip varies. Using a small resolution, say $\Delta t = 0.01$, shows a very random trajectory of the actin tip. There are two problems that arise when we use this small Δt. First, it is difficult to predict where the actin filament is going, as the actin tip seems to drift in all directions. Second, calculating the speed using this small Δt will result in a rather high actin filament speed, characterised by high deviations, which is unrealistic.

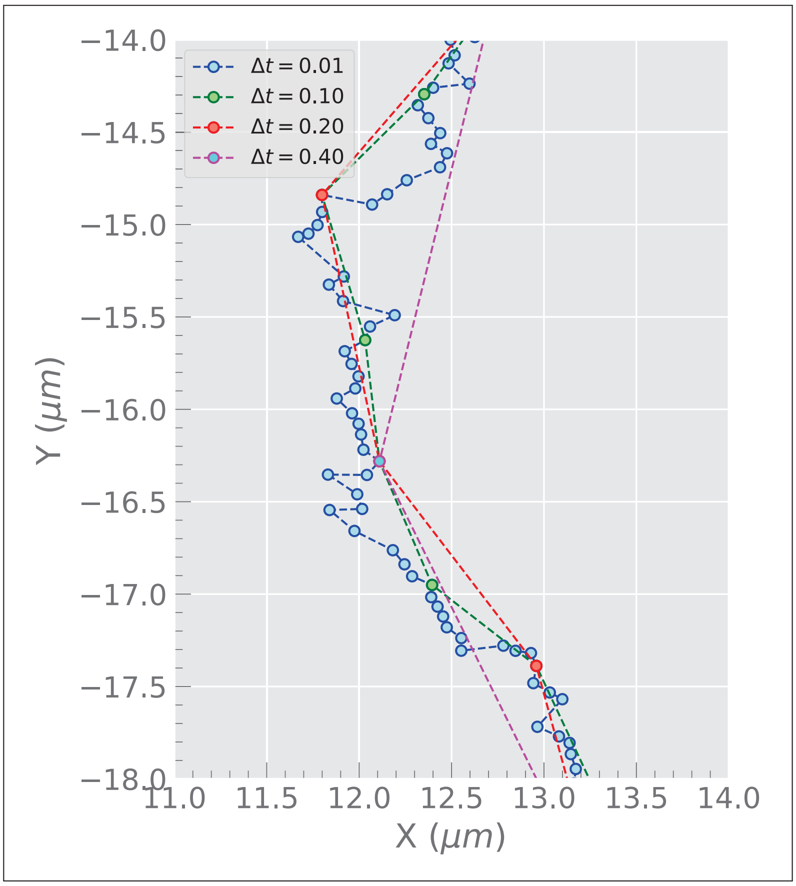


Figure 3.5: Various tip trajectories based on Δt variation.

Figure 3.6 shows various speed calculation results when various Δt are chosen. Around $\Delta t = 0.07$, the speed and deviations of the actin filament seem to settle, and the speed of the actin filament is approximately the same. In this study, we chose

$\Delta t = 0.01$ since at this point the actin filament speed is already settled and in our 5 s simulation, there are still enough data points to calculate the average speed.

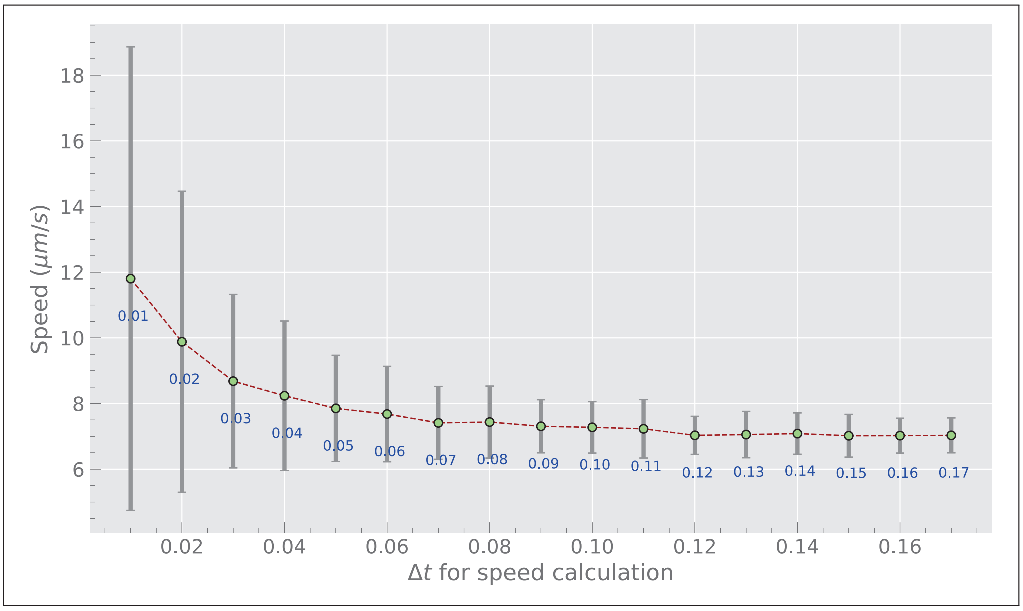


Figure 3.6: Δt used while calculating speed. The error bars here and after represent the standard deviation.

Figure 3.7 shows the velocity of various actin filaments. Here, all parameters were kept constant, but the seed to generate random numbers was changed to simulate various actin filaments. When the random seed of simulation is varied, new actin filaments can be simulated. Various actin filament velocities were observed to be on average approximately the same. All filaments show a speed of approximately 0 $\mu m/s$ in the initiation stage. The speed starts to increase in the transition / intermediate stage, and there is smooth movement in the smooth glide stage, where the speed remains high ($\approx 3-7\mu m/sec.$). Overall, we observed that a high active ratio is required to achieve and maintain a smooth glide of the actin filament. There is a sharp transition between the initiation speed and the smooth glide speed. This implies that the removal of nonfunctional denatured myosin motors will contribute to smooth glide of the actin filament; otherwise, the presence of denatured/defective myosin in the motility assay will cause the actin filament to come to a grinding halt.

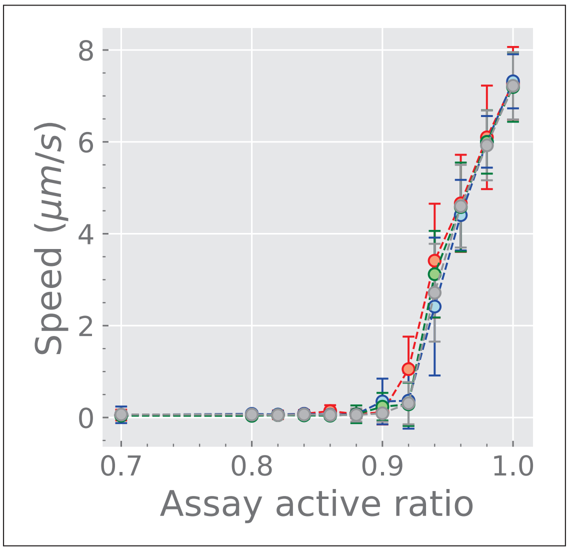
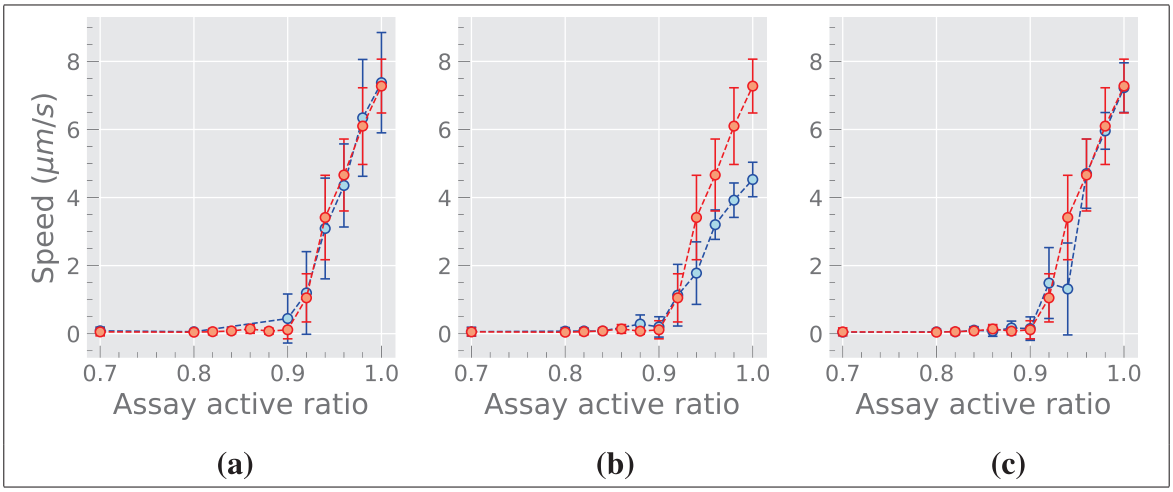


Figure 3.7: Actin filament speed from various filaments.

The gliding onset of the actin filament remains unchanged despite changes in parameters such as motor density, ATP concentration, and actin filament length. Figure 3.8 (a) shows a graph of two actin filament speeds: one at a motor density of 2000 μm^{-2} and the other at a motor density of 3000 μm^{-2}. This difference in motor density does not translate into a significant difference in actin filament speed. In Figure 3.8(b), a low ATP concentration (500 μM) produces a slower actin filament speed, but the gliding initiation position remains the same ($r_{substrate} \approx 0.92$). Similar results are obtained when different actin lengths are used, as shown in Figure 3.8(c). For actin filaments 2 μm and 3 μm, the speed remains the same and the gliding onset is still in the same position. In the simulations in Figure 3.8, the active ratio of the motility assay is the only factor that influences the initiation of gliding of the actin filament.



(a)

(b)

(c)

Figure 3.8: Red: AF speed for AF = 3 μm, ATP = 2000 μM, MD = 3000 μm^{-2}. Blue: **(a)** MD dependence AF = 3 μm, ATP = 2000 μM, MD = 2000 μm^{-2} **(b)** ATP dependence AF = 3 μm, ATP = 500 μM, MD = 3000 μm^{-2} **(c)** length dependence AF = 2 μm, ATP = 2000 μM, MD = 3000 μm^{-2}

In our simulation, we found that the speed of the actin filament is markedly influenced by the set detachment force of the defective myosin. In Figure 3.9, the initiation of glide of the actin filament varies depending on the set detachment force. For the detachment force of 9.2 pN, the onset of glide is at $r_{substrate} = 0.9$. For lower detachment forces of 6.9 pN and 4.6 pN, the initiation of the glide started at $r_{substrate} = 0.8$ and $r_{substrate} = 0.4$, respectively. This effect is similar to that of the load-dependence force-velocity relation studied by others [71], [72]. The higher the load on myosin, the slower the actin filament moves. Here, the set detachment force is directly proportional to the onset of the gliding. Reducing the detachment force by half from 9.2 pN to 4.6 pN reduces the gliding onset from 0.9 to 0.4, which is close to half. This implies that the onset of glide of the actin filament can be improved by making defective myosins dissociate faster. A method such as that studied by *Shen et al.* [73] may serve to improve the speed of the actin filament.

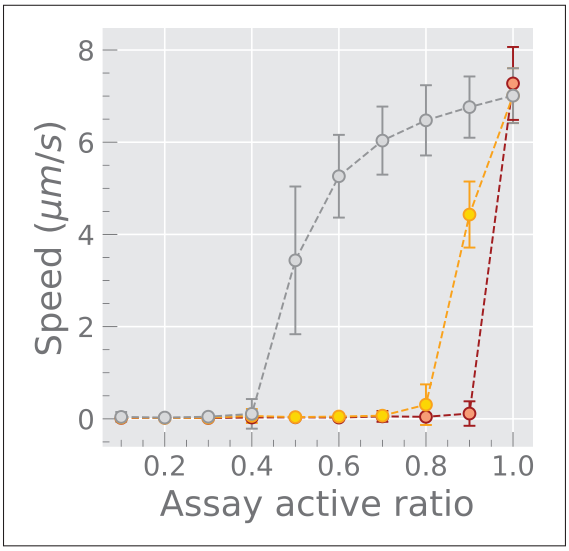


Figure 3.9: Actin speed with various detachment forces. Red:detachforce = 9.2 pN, Lime: detachment force = 6.9 pN, Grey: detachment force = 4.6 pN.

3.1.3 Actin Filament Path Persistence Length

The trajectory plot of Figure 3.10 shows the trajectories of seven actin filaments as the value of $r_{substrate}$ is systematically reduced. The original $\Delta t = 0.01$ of the simulation output is used to make this graph. However, as explained in Section 3.3.3, $\Delta t = 0.2$ and 30% of the data are used during the calculation of L_p.

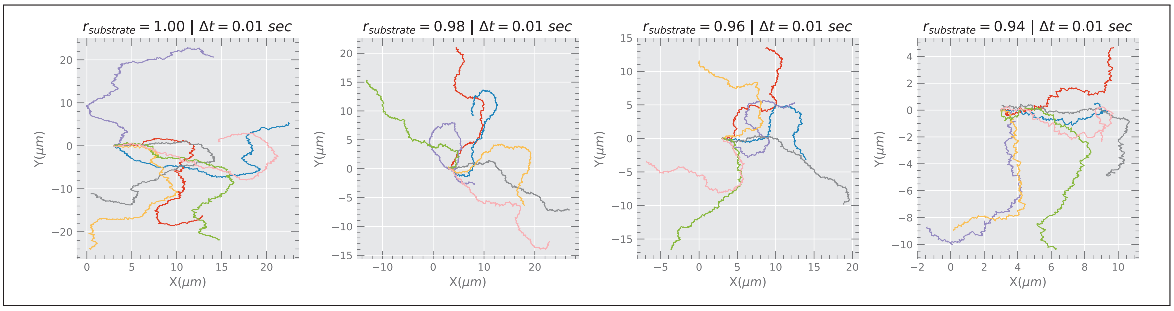


Figure 3.10: Seven actin filament trajectories. From left to right, $r_{substrate} = [1.00, 0.98, 0.96, 0.94]$.

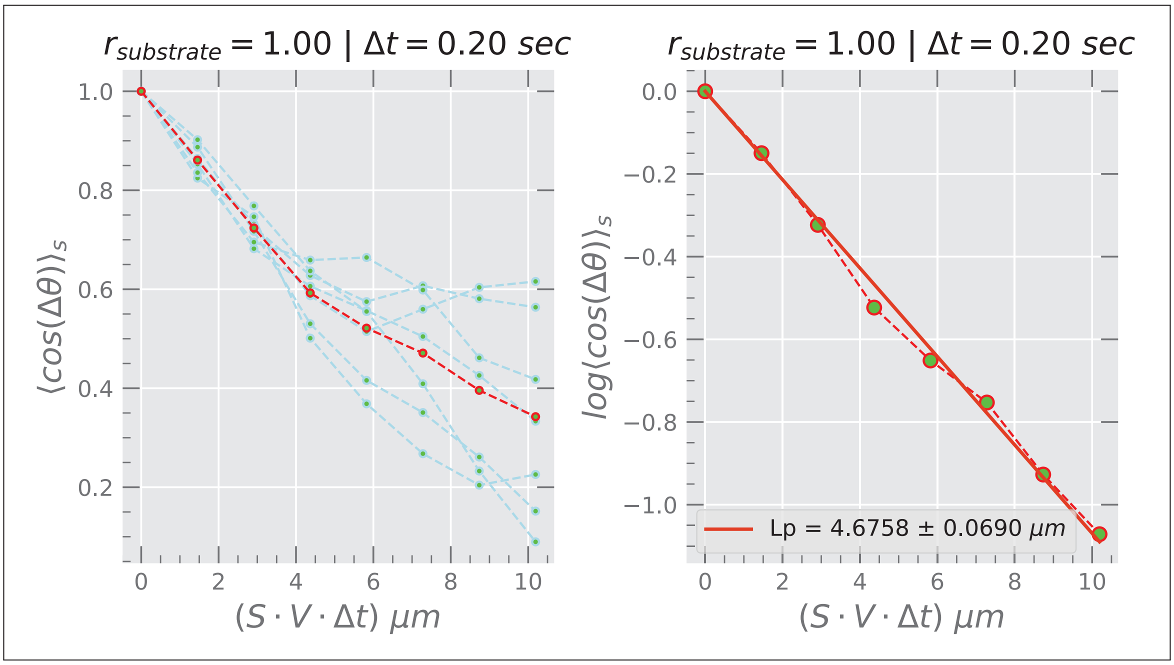


Figure 3.11: Actin filament L_p at $r_{substrate} = 1.00$

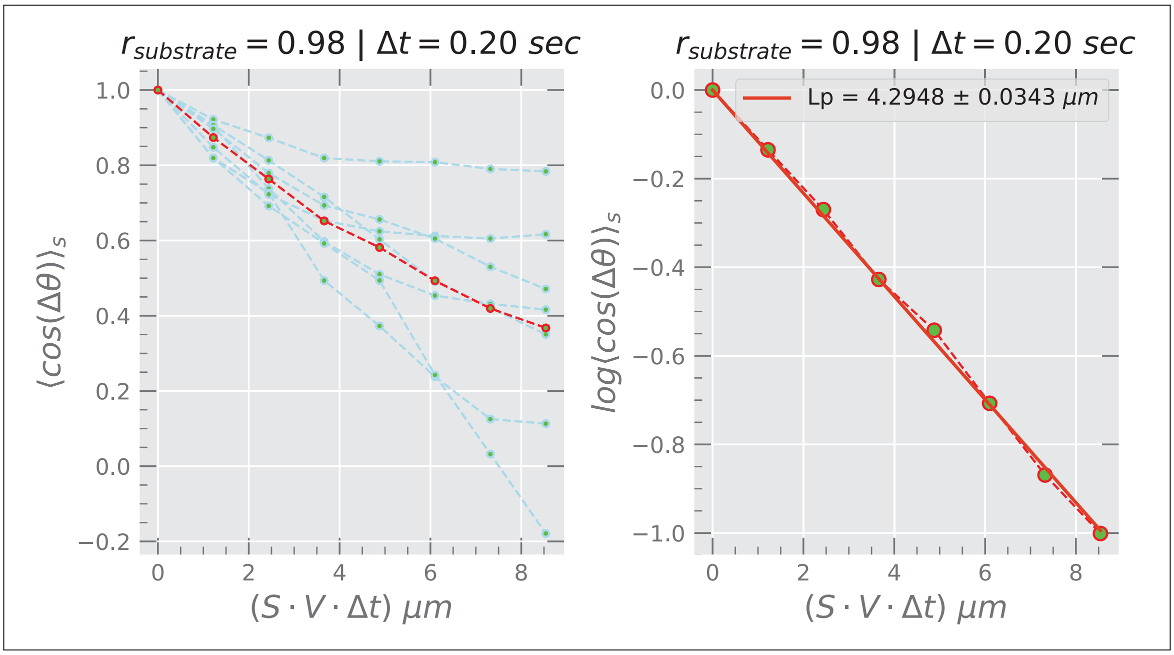


Figure 3.12: Actin filament L_p at $r_{substrate} = 0.98$

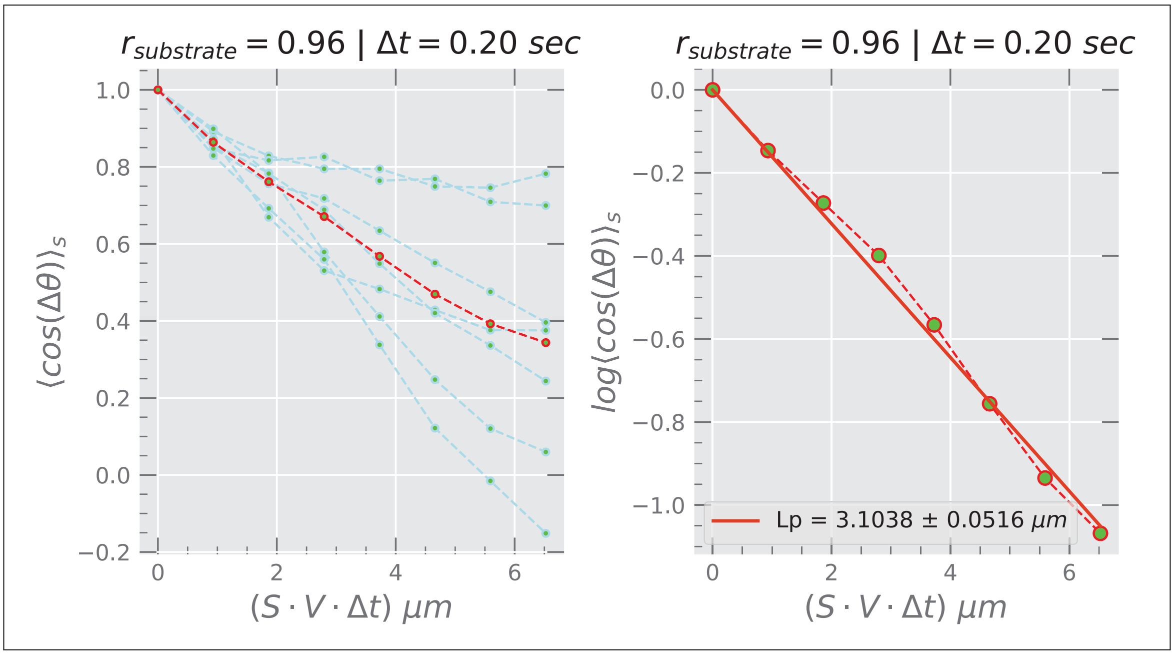


Figure 3.13: Actin filament L_p at $r_{substrate} = 0.96$

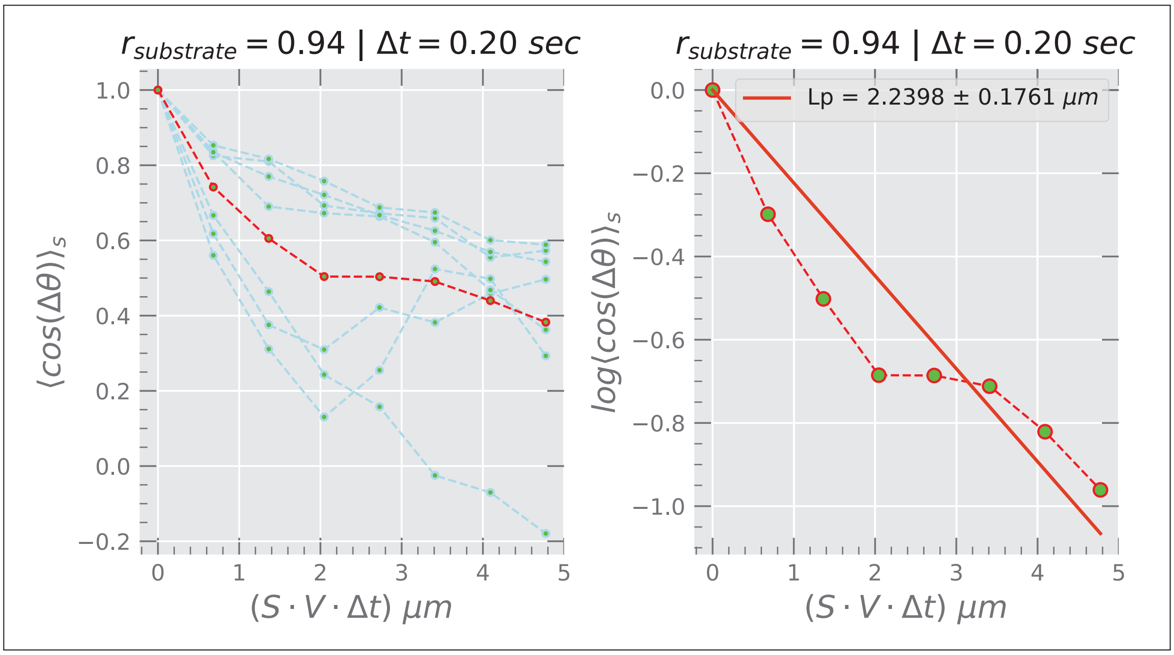


Figure 3.14: Actin filament L_p at $r_{substrate} = 0.94$

Figure 3.15 shows changes in the average L_p as the active motor ratio, $r_{substrate}$, increases from the continuous glide threshold of 0.94 to 1.00. It can be deduced that as $r_{substrate}$ increases, the actin filament L_p improves from 2.2 μm to 4.6 μm. This may be attributed to the high speed of the actin filament with more active motors, which tends to translate the actin filament with fewer wiggling effects.

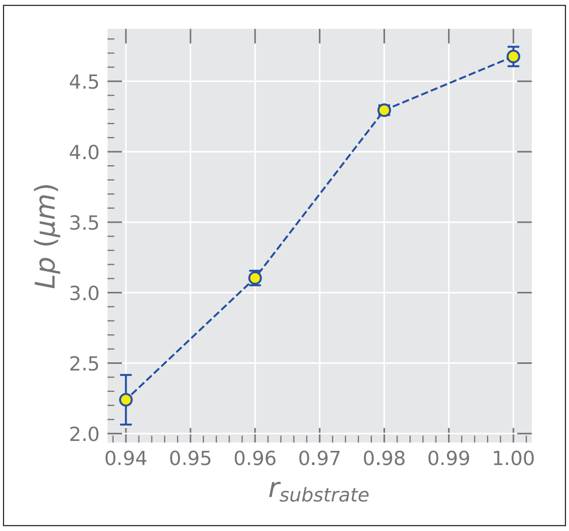


Figure 3.15: Actin filament L_p changes as $r_{substrate}$ increases.

3.1.4 Myosin Binding to the Actin

In this section, binding changes in the in silico motility assay are discussed. Speed changes are also discussed in relation to myosin binding changes.

Binding Changes

To further probe the onset of glide, we plotted the instantaneous binding of myosin to the actin filament as a function of time. As seen in Figure 3.16, the binding myosin behaviour can still be described in three phases. During the initiation phase, defective binding myosins increase rapidly, fluctuate less, and are much more abundant than active myosins. Active binding myosins in this phase fluctuate, but generally stay much lower than that of defective myosin binding. In the transition phase, the binding active myosins generally exceed the binding defective myosins, but there are some instances in which the latter might exceed the former. The number of defective and active myosins bound in this phase fluctuates more than in any other phase. In the glide phase, the binding active myosins are always more binding defective myosins. From this observation, it can be deduced that, for the actin filament to glide, the number of active myosin bindings to the actin filament must exceed that of the binding defective myosin.

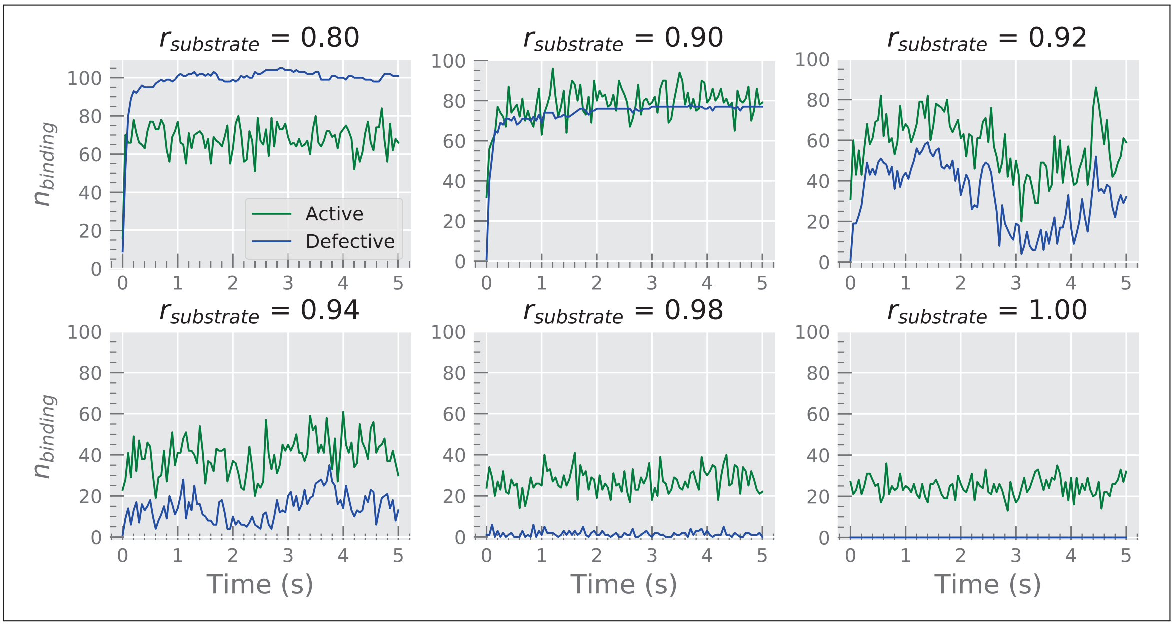


Figure 3.16: Instantaneous binding myosins. ATP = 2000 μM, and motor density = 3000 μm^{-2}

Figure 3.17 shows the average binding myosin in each segment of an actin filament during a 5-second simulation. During the initiation phase, defective myosins that bind to all segments of the actin filament are more common than active myosins. The binding pattern is generally random along the actin filament segments. During the transition phase, the number of defective myosins that bind to every segment is generally less than the number of active myosins. Interestingly, in the transition phase, while the active myosins binding to the segments of the actin filament remain relatively the same, the defective myosins binding to the leading segments are more than the myosins binding around the middle and tail segments of the actin filament. This reinforces the previous discussion of Figure 3.3 and Figure 3.4. In the smooth gliding phase, the defective myosin binding to the actin filament in every segment is lower than that of the active binding myosin, and the segment distribution is uniform.

As long as defective myosins that adhere to the actin filament in every segment are more common than active myosins, glide will not occur. Any segment will tend to move whenever the binding active myosins exceed the binding defective myosins.

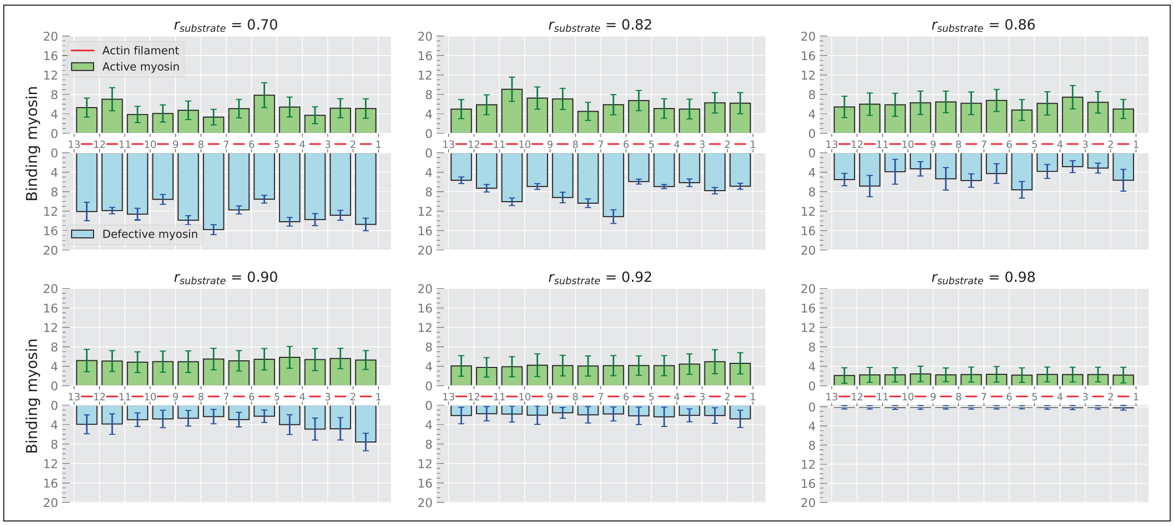


Figure 3.17: Mean binding myosins in each segment of the actin filament.

Binding Myosin and Speed

To understand the relationship between binding myosin and the speed produced by the moving actin filament, we examine these quantities side by side. In our simulation, the introduction of defective myosin into the in vitro motility assay hampers motility almost completely. As seen in Figure 3.18 (left), the actomyosin assay containing defective myosin $\geq 10\%$ does not translate the actin filament. At this point, the actin filament speed is $\approx 0\ \mu m/s^2$ and the active binding myosins are much lower than the defective binding myosins (Figure 3.18 (middle)). From the onset of glide ($r_{substrate} \approx 0.92$), the minimum procession speed was observed to be $0.2 \pm 0.75\ \mu m/sec.$. Active binding myosins are ≈ 50, while defective binding myosins are half that of active myosins, ≈ 25. The maximum speed was observed to be $7.25 \pm 0.75\ \mu m/sec.$ during a smooth glide. The smooth glide speed of $7 \mu m/sec.$ is much higher than that achieved in experiments such as *Hanson et al.* [74]. However, this discrepancy may be explained by a difference in temperature when performing an experiment [75]. Furthermore, in an experiment to investigate the motion of F-Actin and its attachment to surfaces, Fig. 5 of *Winkelmann et al.* [76] shows how various temperatures can influence the velocity of the actin filament. In all phases, except at $r_{substrate} = 1.0$, the ratio of active myosin binding to the actin filament is much lower than the active myosins available in the motility assay (Figure 3.18 (right), Figure 3.21).

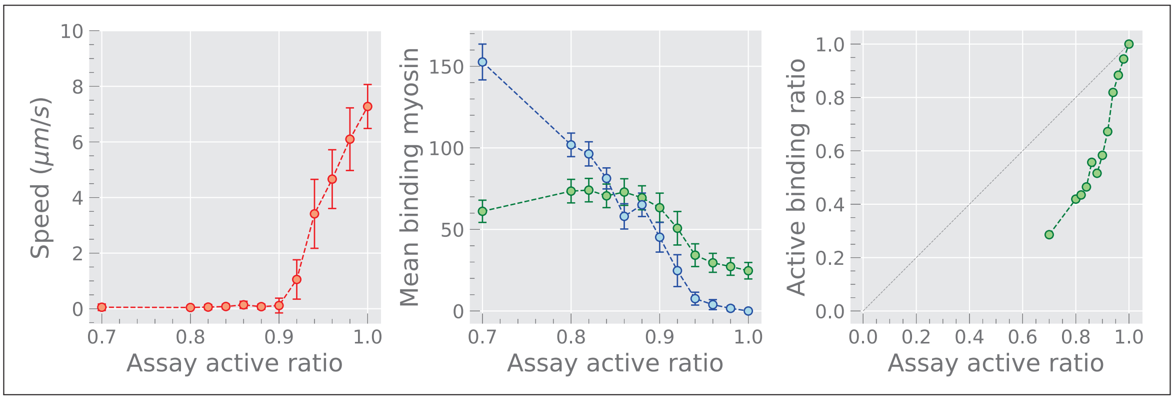


Figure 3.18: Actin speed and binding myosin changes. (a) AF speed changes with increase in active myosin ratio, (b) binding active (green) and binding defective (blue) myosin number, (c) changes in active binding ratio

To determine how binding myosins change when subjected to different conditions, we simulated the actin filament under different motor densities and ATP concentration. Figures 3.19 and Figure 3.20 show how the speed of the actin filament and the associated binding motor change when the actin filament is subjected to a lower myosin density of 2000 μm^{-2} and a lower ATP of 500 μM, respectively. The speed of the actin filament is similar to that of the previous higher myosin density of 3000 μm^{-2}. However, the average number of myosin binding to the actin filament is lower for the motor density of 2000μm^{-2}. At $r_{substrate} = 1.00$ and at a myosin density of 2000 μm^{-2}, the average number of myosin binding to actin is the lowest and, as a result, the actin filament is more likely to detach at a motor density of 2000 μm^{-2} than at a motor density of 3000 μm^{-2}.

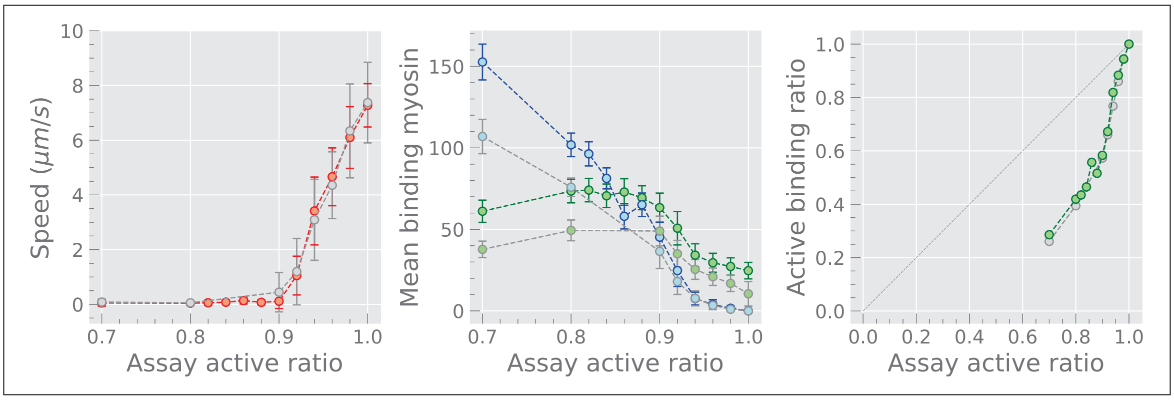


Figure 3.19: Actin speed and binding myosin changes at MD 2000.

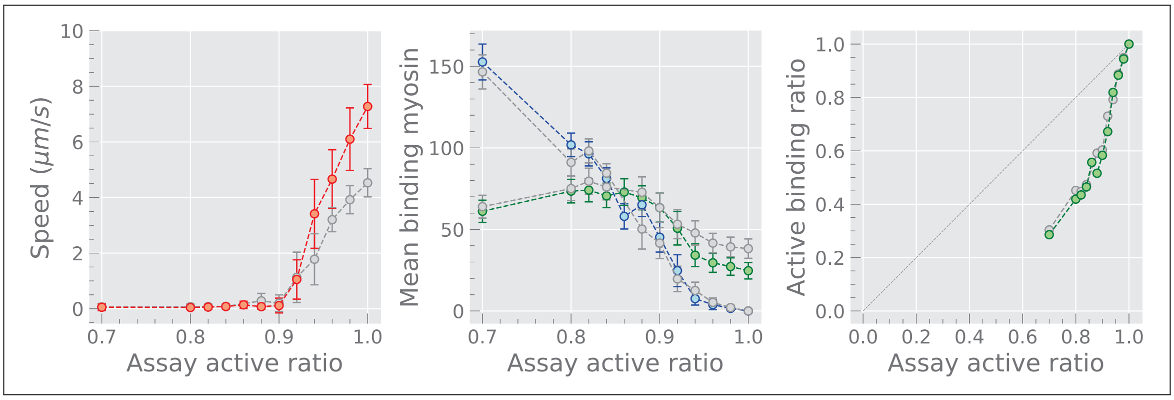


Figure 3.20: Actin speed and binding myosin changes at ATP 500.

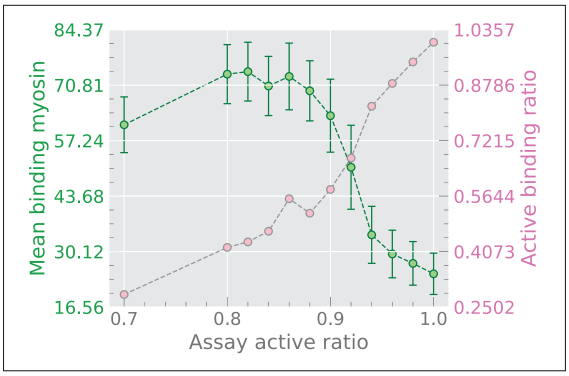


Figure 3.21: Binding myosin, binding ratio, lifetime comparison. Motor density = $3000\mu m^{-2}$, ATP = $2000\mu M$

The actin speed and the trend of the binding ratio remain the same, except for ATP 500 μM, where the speed is significantly lower. For both 2000 μM and 500 μM, the average number of myosins binding to the actin filament remains the same until the start of smooth glide of the actin filament. At this point, the number of binding myosins for 500 μM is greater than that for 2000 μM. The increase in the number of active myosins binding at low ATP is plausible due to the dependence of myosins on ATP, as discussed in *Lymn et al.* [77]. Since myosin can bind to the actin filament or ATP [78], when there is a deficiency in ATP, more myosin will remain bound to the actin filament.

The amount of force required to detach a defective myosin influences the number of active myosins that adhere to the actin filament. In Figure 3.22, various detachment forces for defective myosin were used (4.6 pNpN 6.9 pN and 9.2 pN), while the detachment force for active myosin remained the same in all simulations (9.2 pN). For the defective detachment force of 4.6 pN 6.9 pN, and 9.2 pN, the number of myosin binding activity increases linearly until the onset of glide of $r_{substrate}$ = 0.4, $r_{substrate}$ = 0.8, and $r_{substrate}$ = 0.9, respectively (Figure 3.22**(a)**). At higher detachment forces of the defective myosin, a higher active ratio is required for the motility assay to have a higher active ratio, and for the movement of the actin filament, a higher active binding ratio is required. The binding ratio of active myosins to actin filament improves as the detachment force of defective myosin is reduced (Figure 3.22**(b)**). This finding is similar to the behaviour observed by *Cheng et al.*[71]. Defective myosin increases the load on myosin, and more myosins are required to bind to the actin filament to overcome this resistance.

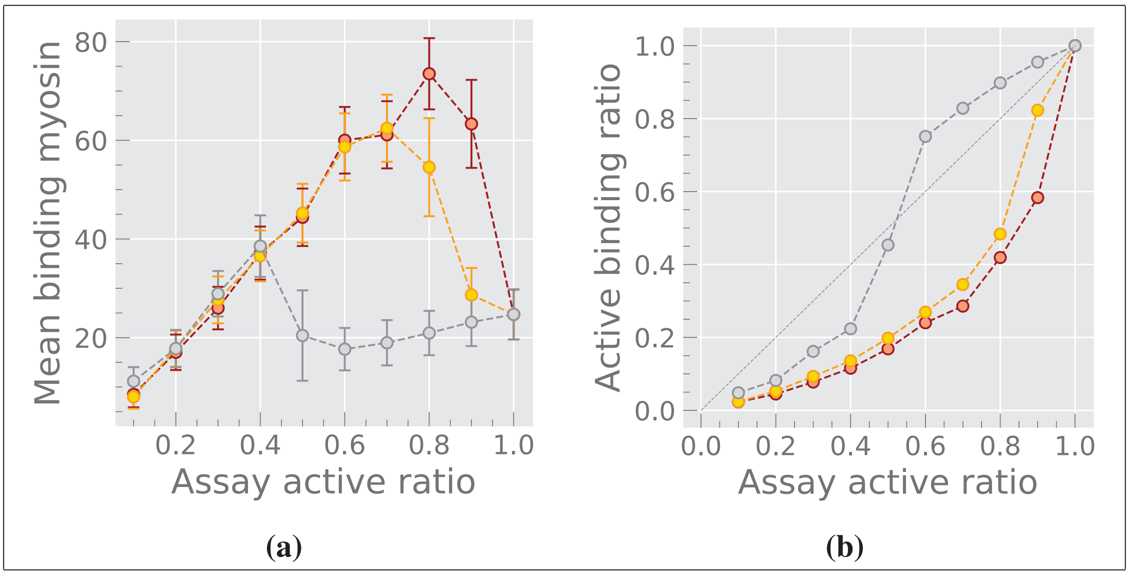


Figure 3.22: Binding myosins and the ratio of various defective myosin detachment forces. Red:detach force = 9.2 pN, Lime: detach force = 6.9 pN, Grey: detach force = 4.6 pN. **(a)** active binding motors with various detachment forces, **(b)** active binding ratio with various detachment forces.

To elucidate the relationship between actin filament speed and binding myosin, we examined the correlation between these parameters. As seen in Figure 3.23, the number of defective myosins binding to the actin filament is highly correlated with the number of active myosins binding to the actin filament. The defective motors were modelled to stick to the actin filament once they were bound, unless overcome by a higher force. Active myosins bind and unbind according to the hydrolysis cycle. When the binding defective myosin increases, the load on the myosin increases, and the number of binding active myosins also increases. It is plausible that the number of active myosins binding to the actin filament increases/readjusts to counter the load and thus tends to favour ADP release, and hence, active myosins stay bound to the actin filament for a longer time.

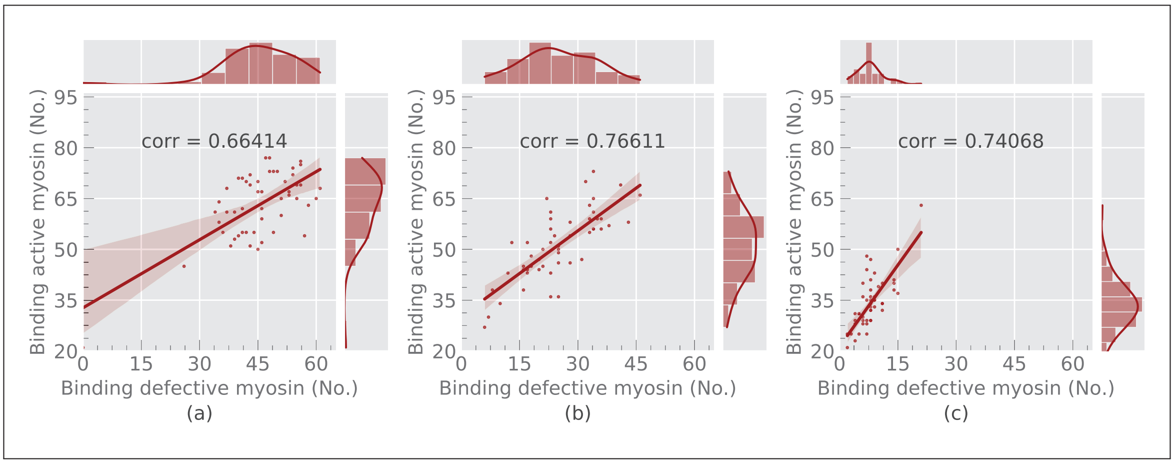


Figure 3.23: Correlation between binding defective and active myosin. (a) $r_{substrate}$ = 0.90, (b) $r_{substrate}$ = 0.92, (c) $r_{substrate}$ = 0.94. ATP = 2000 μM, and motor density = 3000 μm^{-2}

Next, we checked the correlation between the binding of active myosin and the speed of the actin filament. In Figure 3.24, there is no correlation between the number of active myosin binding and the speed. The speed of the actin filament does not necessarily increase with increasing number of binding myosins. As explained in Figure 3.23, the dramatic increase in active myosin binding occurs when the actin filament becomes stuck due to the accumulation of defective myosin, and then active myosin increases to counteract stall. Once the stall has been overcome, the speed of the actin filament does not increase. This implies that the increase in the speed of the glide of the actin filament is not related to an increase in binding active myosin and vice versa. However, as observed in the correlation between the ratio of active myosin binding and the speed of the actin filament in Figure 3.25, there is a faint correlation between the ratio of active myosin binding and the increase in speed at the beginning of glide ($r_{substrate} \approx 0.92$). This suggests that increasing the number of active myosins that adhere to the actin filament during the onset of glide increases the speed.

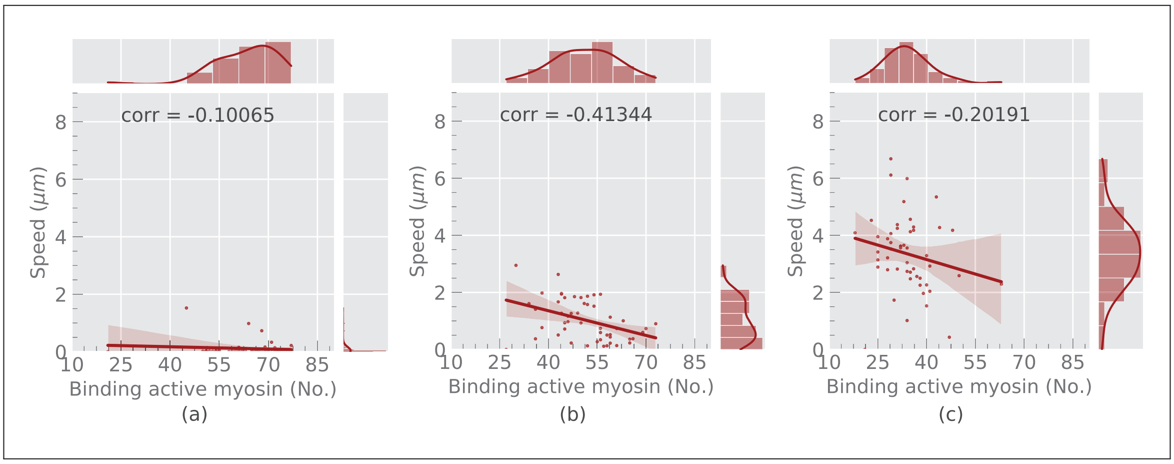


Figure 3.24: Correlation between binding active myosin and actin filament speed. (a) $r_{substrate}$ = 0.90, (b) $r_{substrate}$ = 0.92, (c) $r_{substrate}$ = 0.94. ATP = 2000 μM, and motor density = 3000 μm^{-2}

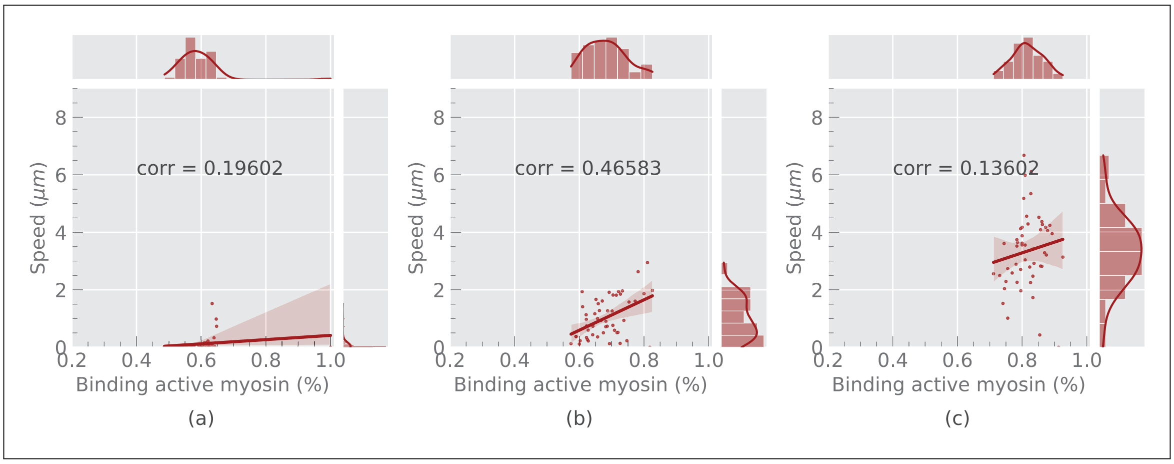


Figure 3.25: Correlation between the ratio of binding active myosin and the actin filament speed. (a) $r_{substrate} = 0.90$, (b) $r_{substrate} = 0.92$, (c) $r_{substrate} = 0.94$. ATP = 2000 μM, and motor density = 3000 μm^{-2}

To check whether the number of myosins bound in the simulation is probable, we calculated the number of defective myosins that can bind to the actin filament. The number of defective myosins bound was selected for simplicity. Figure 3.26 shows an actin filament surrounded by randomly distributed myosin motors.

$$Binding\ myosins = (1 - r_{substrate}) \times \rho \times L_{AF} \times 2w, \tag{3.1a}$$

$$\begin{aligned} &where \\ r_{substrate} &: Active\ motor\ ratio \\ L_{AF} &: Length\ of\ actin\ filament \\ \rho &: Myosins\ density \\ w &: Distance\ within\ which\ myosin\ can\ bind\ to\ actin \end{aligned}$$

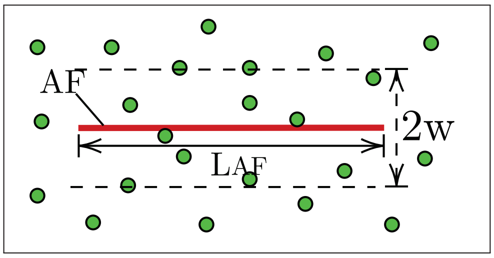


Figure 3.26: Possible number myosins that can bind to the actin filament. Red: actin filament, Green dot: myosin

To estimate the lifetime of binding, the fraction of time that a defective myosin stays attached to the actin filament, it is assumed that the defective motor undergoes a repeated cycle of binding and unbinding:

$$T_{off} = \frac{1}{ka} \tag{3.2a}$$

$$kVT_{on} = Fd;\ T_{on} = \frac{Fd}{kV} \tag{3.2b}$$

$$\frac{T_{on}}{T_{on}+T_{off}} = \frac{\frac{Fd}{kV}}{\frac{Fd}{kV}+\frac{1}{ka}} \tag{3.2c}$$

$$= \frac{Fd}{Fd+\frac{kV}{ka}} \tag{3.2d}$$

Defective binding myosin can be expressed as:

$$\frac{Fd}{(Fd)+(kV/ka)} \times (1-r_{substrate}) \times \rho \times L_{AF} \times 2(Fd/k), \tag{3.3}$$

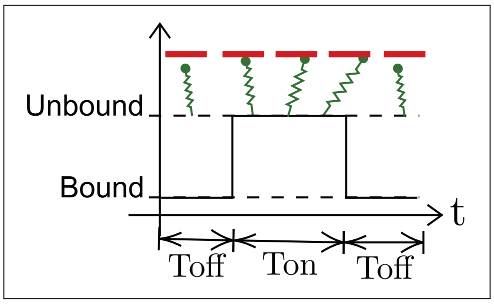


Figure 3.27: Estimating time taken by a defective myosin to stay bound to the actin filament.

$$\begin{aligned}
&where\\
&\quad ka : Binding\ rate\\
&\quad k : Spring\ constant\\
&\quad Fd : Detachment\ force\\
&\quad V : Gliding\ speed\ of\ the\ actin\ filament
\end{aligned}$$

Figure 3.28 shows the fit between the simulation output and the estimate of Equation (3.3). The results of independent mathematical calculations fit well with the simulation results. This estimation formula does not apply to the binding of active myosin to the actin filament, since more complex factors such as the hydrolysis cycle have to be taken into consideration.

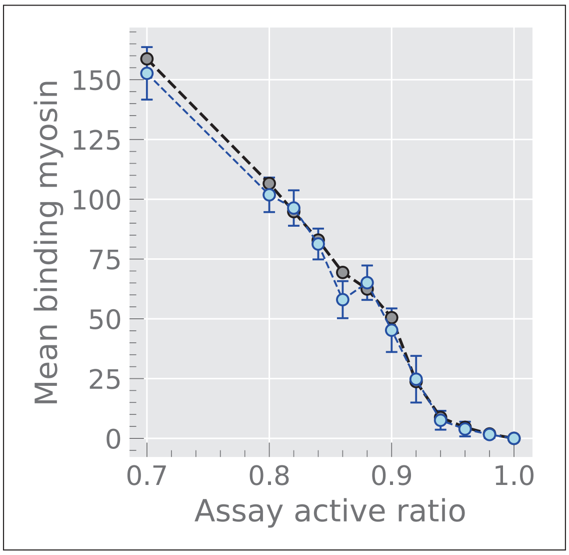


Figure 3.28: Predicting binding defective myosin. Blue: simulation, Black: Equation (3.3).

3.1.5 Myosin Lifetime

Figure 3.29 shows the average time myosin remains bound to the actin filament, hereafter referred to as lifetime. The lifetime of active and defective myosin binding generally decreases as the active myosin ratio of the motility assay gradually increases from 0.7 to 1.0. For active myosin, the lifetime is longer in the initiation phase and shorter in the glide phase. In the initiation phase, the load on the active myosin is larger due to the presence of binding defective myosins that remain stuck to the actin filament for an average time of around 4.4 seconds, almost throughout the 5 seconds of simulation. At the same time, active myosin binds to the actin filament for a period of a little more than 0.012 seconds. This load encourages the release of ADP and hence facilitates binding. Several defective myosins bind to the actin filament without detaching, causing the actin filament to stay in situ. The load on myosin gradually eases during the transition phase and is lowest during the glide phase. The lifetime of the binding myosin adjusts according to the applied force and this is consistent with the load dependence described by *Veigel et al.* [72], [79], [80]. The lifetime of defective myosin shown in Figure 3.29 (right) is much longer in the initiation phase and rapidly decreases to ≈ 0 during the glide phase. This underpins the importance of the purification techniques of the motility test discussed by *Kron et al.* [69].

Figure 3.30 shows the lifetime of defective myosin in detail. As $r_{substrate}$ is reduced below the gliding outset, the lifetime of the defective myosin increases as the simulation time increases. Therefore, the defective lifetime tends to infinity as the simulation time increases, similar to Equation (3.4). To validate the results, the red dot dotted line in Figure 3.30(right) uses Equation (3.2b) and the speed of the actin filament to estimate the lifetime of the defective myosin. The trend of the estimated life is similar to the simulated lifetime.

$$\lim_{\tau_{simulation}\to\infty} \tau_{life} \approx \infty \tag{3.4}$$

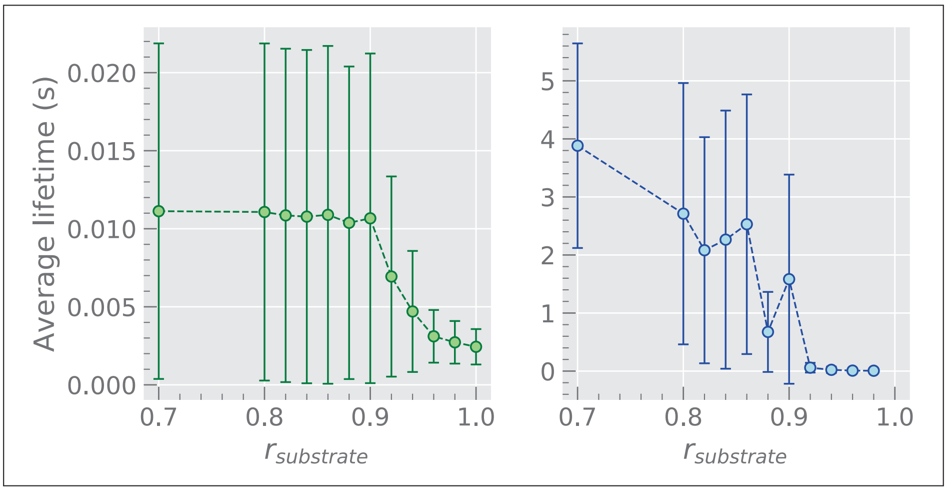


Figure 3.29: Comparing myosin lifetimes. Green: active myosin, Blue: defective myosin.

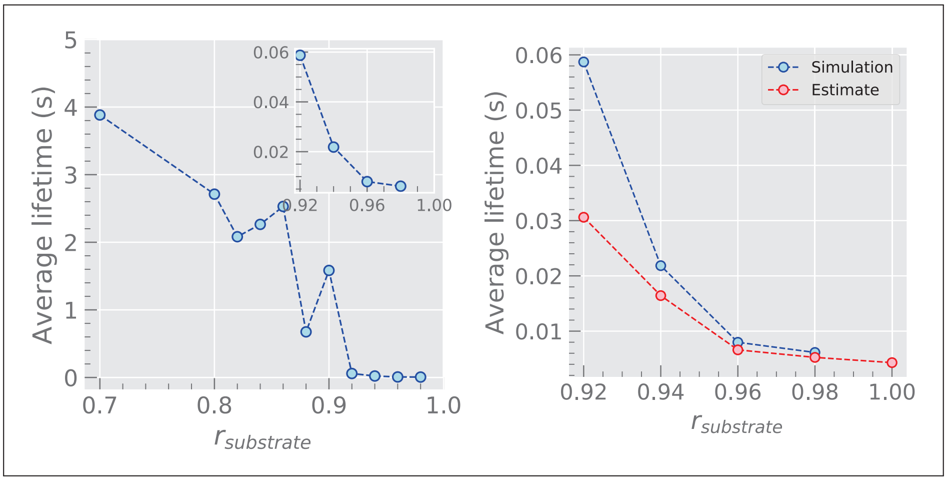


Figure 3.30: The defective myosin lifetime during continuous gliding. The estimate was done using Equation (3.2b).

Figure 3.31 shows the lifetime distribution of active and defective binding myosins, respectively. On the one hand, the lifetime of the binding active myosins follows the Poisson distribution because the binding behaviour is independent and the binding rate is constant. On the other hand, the lifetime distribution of defective binding myosin is random during the initiation phase and changes to Poisson during the glide phase. This suggests that smooth glide is preferred when various defective myosins do not bind to the actin filament at the same time. Continuous glide could also be favoured when the lifetime of defective myosin binding is reduced. This may be achieved by increasing the ionic strength of the buffer.

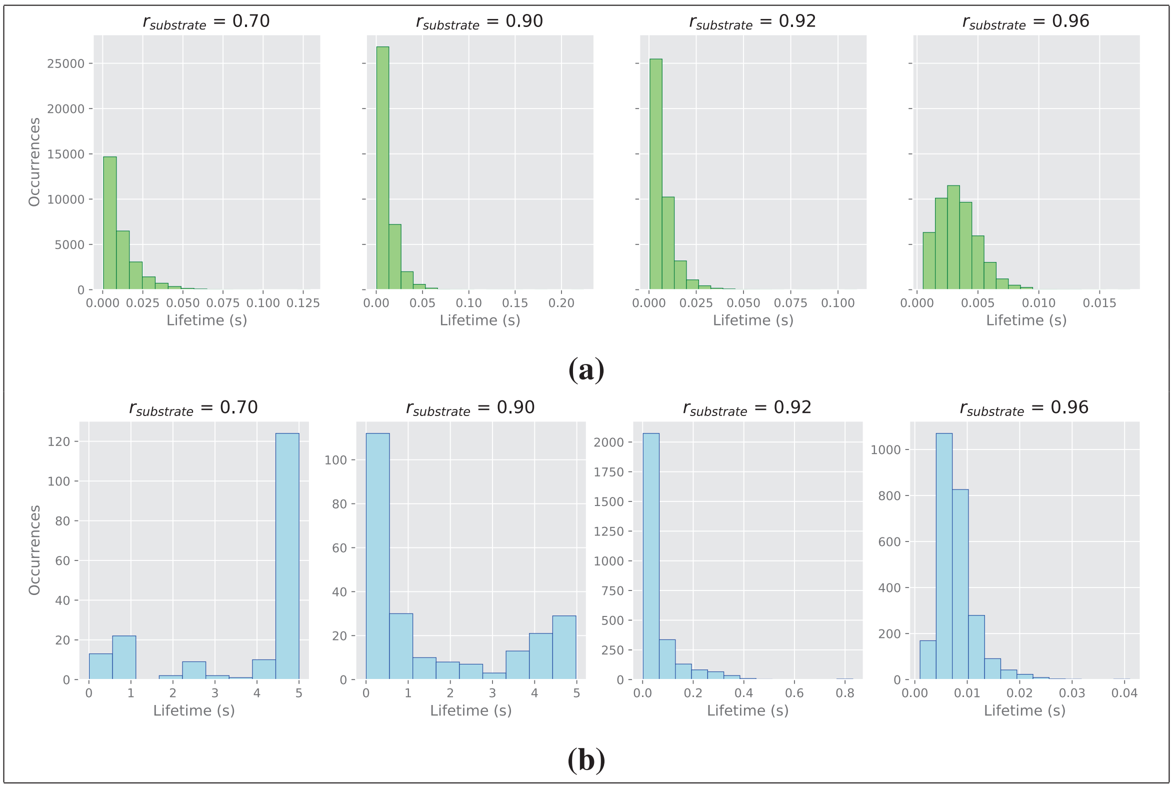


Figure 3.31: **(a)** Lifetime histogram for binding active motors. **(b)** Lifetime histogram for binding defective motors

To further investigate the phenomenon of myosin load dependence, we checked the lifetime of active myosin with and without the load dependence function. In Figure 3.32, the lifetime of the active myosin binding remained the lowest (≈ 0.003 *sec.*) when the simulation was done without the load-dependence function. This justifies that the longer lifetime recorded by active myosins during the initiation phase is due to the contribution of the load-dependence Equation (2.8).

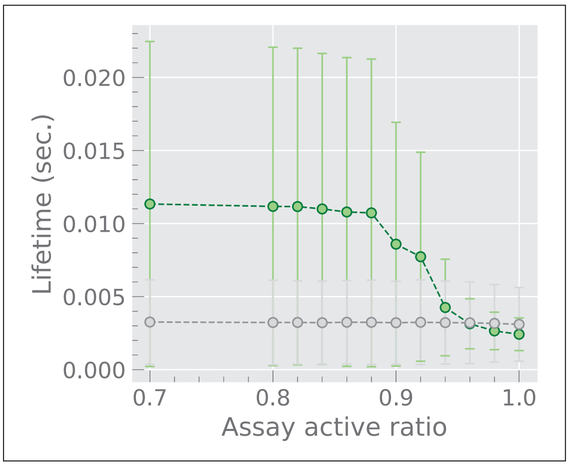


Figure 3.32: Green: active binding myosin with load dependence, grey: active binding myosin without load dependence. Assay active ratio $= r_{substrate}$. The active myosin lifetime data is from another set of simulation similar to Figure 3.29(left)

Figure 3.33 shows the lifetime of active binding myosin under different defective detachment force conditions. Before the load caused by the binding defective myosin is overcome, the lifetime of the active motor remains the highest (0.011 *sec.*).

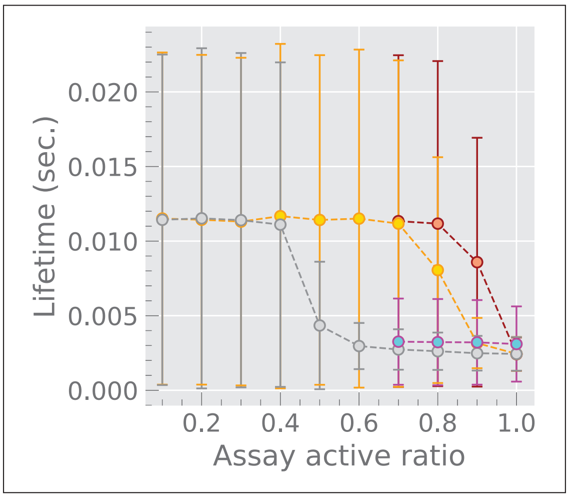


Figure 3.33: Comparing lifetime in various defective myosin detachment force conditions. Red: detachment force = 9.2 pN, Lime: detachment force = 6.9 pN, Grey: detachment force = 4.6 pN, Blue: without load dependence function

3.2 Actin over Active and Defective Myosin: 1D Mathematical Model

In this section, the results of the 1D mathematical model are discussed in detail, in connection to the 3D simulation above.

3.2.1 Model Predictions

Figure 3.34 shows the possible solutions for the velocity given the various forces applied in our mathematical model.

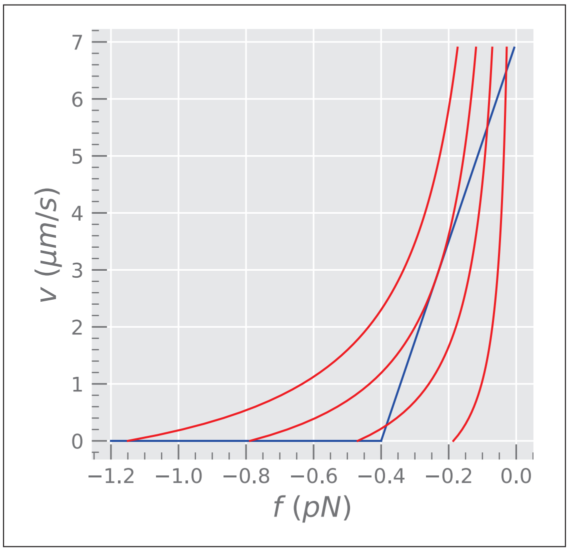


Figure 3.34: Model possible solutions for myosin

From the schematic of Figure 3.35, which is based on the mathematical models derived in Section 2.2, stationary to mobile prediction can be obtained from the intersection of the force-velocity and impedance (active ratio) plots. In Figure 3.35 (left), at the stationary point, the red impedance graph #4 (obtained from Equation (2.12b)) intersects the blue force-velocity graph (obtained from Equation (2.10)) at $v = 0\ \mu m/s$. The impedance plot #3 has only one speed solution (one intersection). From the impedance plots between #3 and #2 there are two solutions. The impedance plots around #1 have one high-speed solution. This shows a discontinuous stationary-motile transition of actin-based molecular shuttles, as seen in Figure 3.35(right). The slope of the force-velocity graph is mostly steeper than that of the impedance graph, that is, the ratio $\frac{-k\tau_1 v_{max}}{2f_{rupt}} > 1$. Therefore, the angle of slope at the intersection of these graphs can be used to describe the stationary to motile transition of actin filaments. Depending on the active ratio of myosin motors, the motility of the actin-based molecular shuttle makes sharp transitions from stationary to motile, compared to the molecular shuttles based on microtubules [81]. This model can be applied to microtubule-based molecular shuttles to elucidate motility.

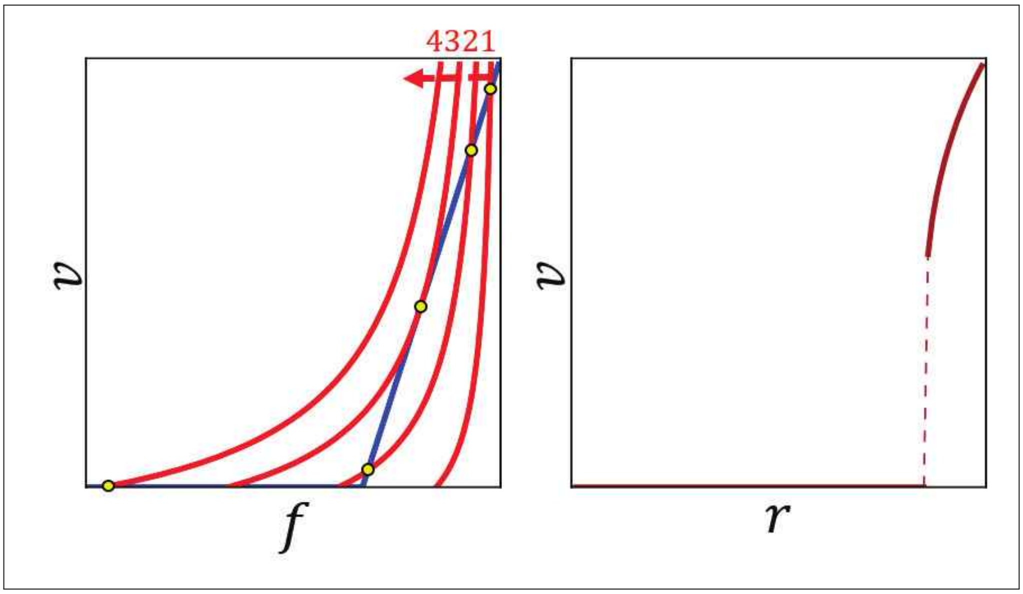


Figure 3.35: Model prediction schematic. Red: active ratio (representing impedance), Blue: force-velocity relationship

For the case of $\frac{-k\tau_1 v_{max}}{2f_{rupt}} > 1$, the three cases described above can be summarised as follows. In the first case, $\frac{\rho_d}{\rho_a}$ is small and the red curve #1 of Figure 3.35(left) has one intersection that gives one solution equal to the positive solution of Equation (2.13e). In the second case, the red curve #2 intersects the blue force-velocity line at two points, providing two solutions of Equation (2.13e). In the third case, the red curve #4 does not intersect the force-velocity line and there is no motile solution.

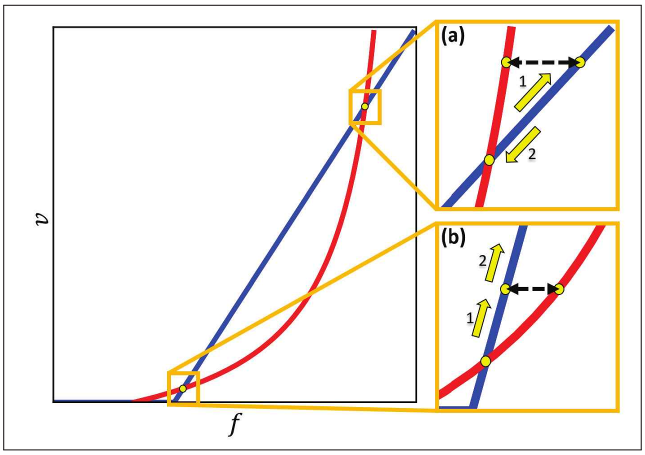


Figure 3.36: Stability of the solutions

In Figure 3.36,

In the mathematical model, we solved for an active ratio of 0.8 to 0.9. The blue line of Equation 1 represents the speed, and the red line of Equation 4 represents the impedance. There are three cases of solutions to this mathematical model. In the first case, the impedance is above the critical value and there is no motile solution. In the second case, the impedance is at the critical value, and there is only one motile solution. In the third case, there are two motile solutions, but only one is stable. When the low-speed solution is considered, if the actin filament fluctuations cause its speed to increase, as shown in the figure, the impedance reduces, causing the actin filament speed to accelerate even further to a higher speed. The filament does not return to its original speed, which means that it is unstable. With a high-speed solution, if the speed of the actin filament increases as a result of thermal fluctuations, the corresponding impedance increases, and the actin filament speed decelerates back to its original speed. Therefore, this speed is stable.

In the 1D mathematical model, the active ratio $r_{substrate}$ is the single important factor that influences the onset of glide of the actin filament and speed. An active ratio of more than 80% is required to maintain a continuous glide movement of the actin filament. The results for the 1D mathematical model were similar to those of the 3D simulation discussed previously.

3.2.2 Model Applications

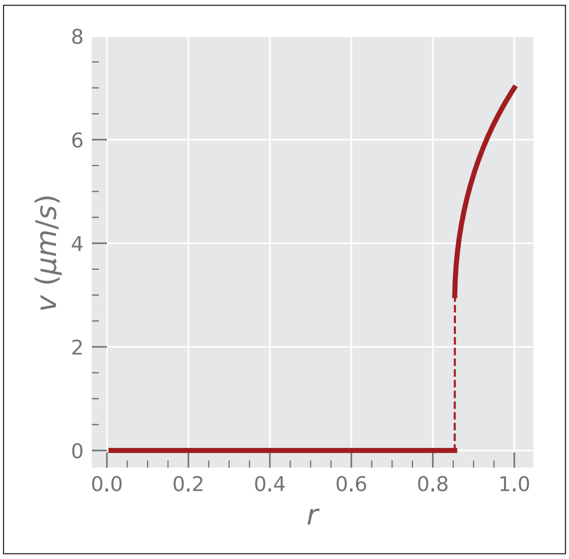


Figure 3.37: The predicted speed of the actin filament.

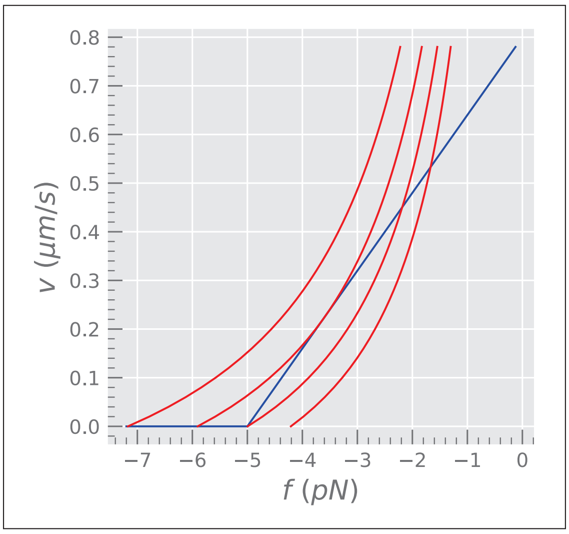


Figure 3.38: Prediction of changes in active ratios of vs. $f - v$ relationship of kinesin.

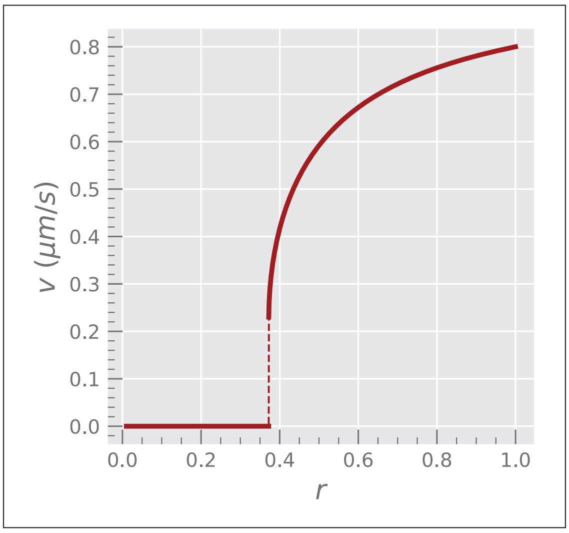


Figure 3.39: Kinesin speed prediction for the mathematical model

The mathematical model solution for the active motor ratio can be summarised in three cases. As illustrated in Figure 3.35 (left), the first case does not have a solution. The red $r_{substrate}$ curve does not intersect the blue $f - v$ impedance curve. The second case has one solution, where the $r_{substrate}$ curve intersects the $f - v$ line at one point, which is the critical active motor ratio of 0.85. The third curve intersection has two solutions, and we must pick the stable solution.

Figure 3.40 shows the point at which the glide of the actin filament started.

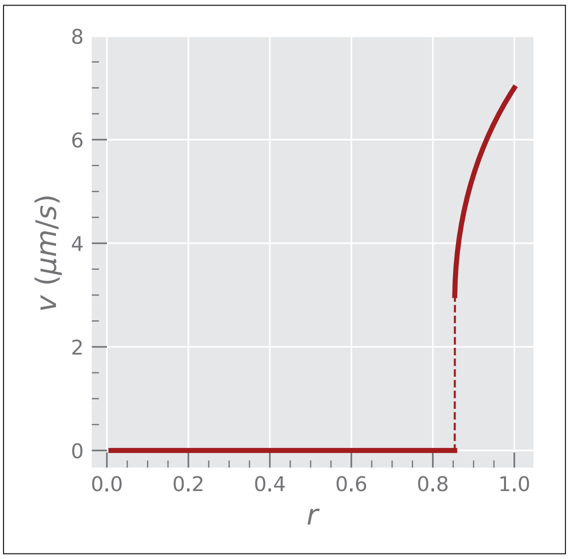


Figure 3.40: Model gliding onset for $f_{rupt} = 9.2pN$. r represent the $r_{substrate}$.

The plot in Figure 3.41 shows the results of the velocity versus the active motility after using various detach forces for defective motors.

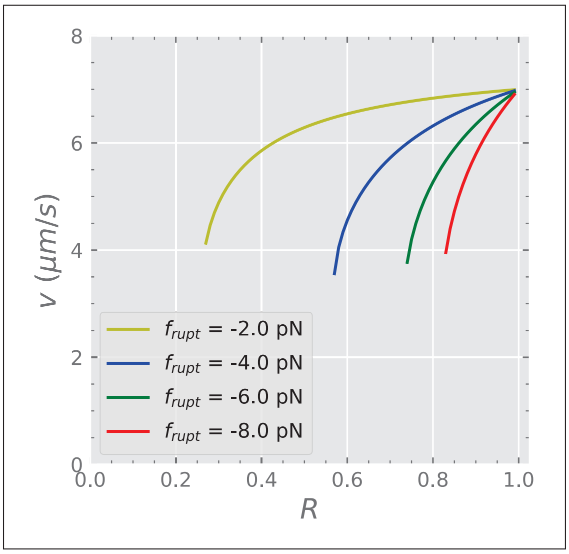


Figure 3.41: Model gliding onset for various detachment forces (f_{rupt}). R represent the $r_{substrate}$.

3.3 Actin under External Force: 2D Simulation

In this section, results of the simulation concerning the controllability of the actin filament using an electric field are discussed in detail.

3.3.1 Effect on the Trajectories

Figures 3.42 and Figure 3.43 show the trajectories of the actin filament after applying various magnitudes of positive and negative force fields. In both figures, when no force is applied, the actin filament moves around randomly, making unpredictable trajectories. When the negative force field was applied (Figure 3.42), the actin filament trajectories eventually took a U turn away from their initial direction. As the magnitude of the external force field increased, the trajectories took sharper U turns, and the alignment between one trajectory and the next became almost packed together in parallel (at $F = -1.0 pN/\mu m$). The application of a positive external force field (Figure 3.43) showed similar results, the only difference being that the trajectories continued straight without taking a U-turn. As the positive external force field was increased, the trajectories loci became predictable, close together, and more aligned on a straight line. This feature is useful in the delivery of directed analytes.

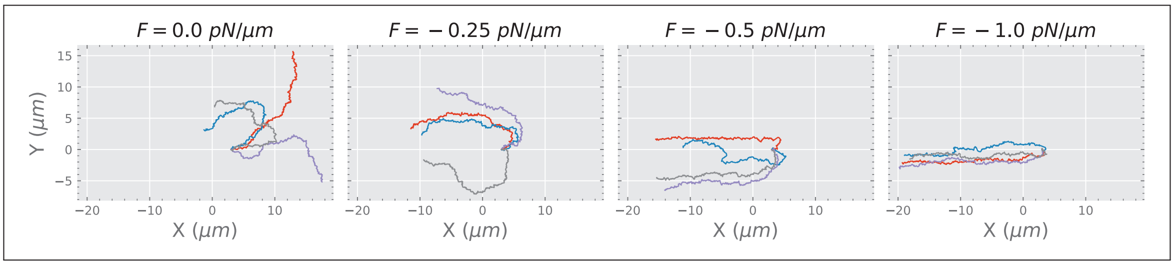


Figure 3.42: Negative force field trajectories

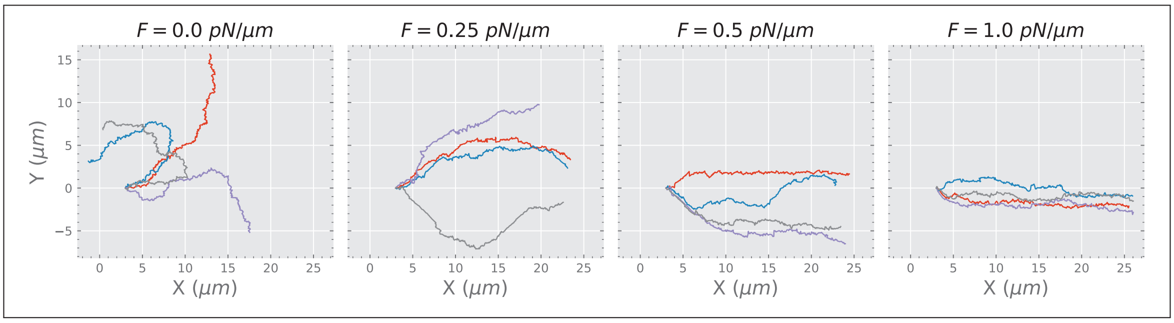


Figure 3.43: Positive force field trajectories

Figure 3.44 shows the effectiveness of applying an external force to actin filaments. For both positive and negative force fields, the trajectories of actin filaments become more predictable. We found that, in addition to making it possible for the actin filament trajectories to align, the application of an external force field increases the distance travelled by the actin filament. These results are important in fast drug delivery applications, as will be discussed later in this section.

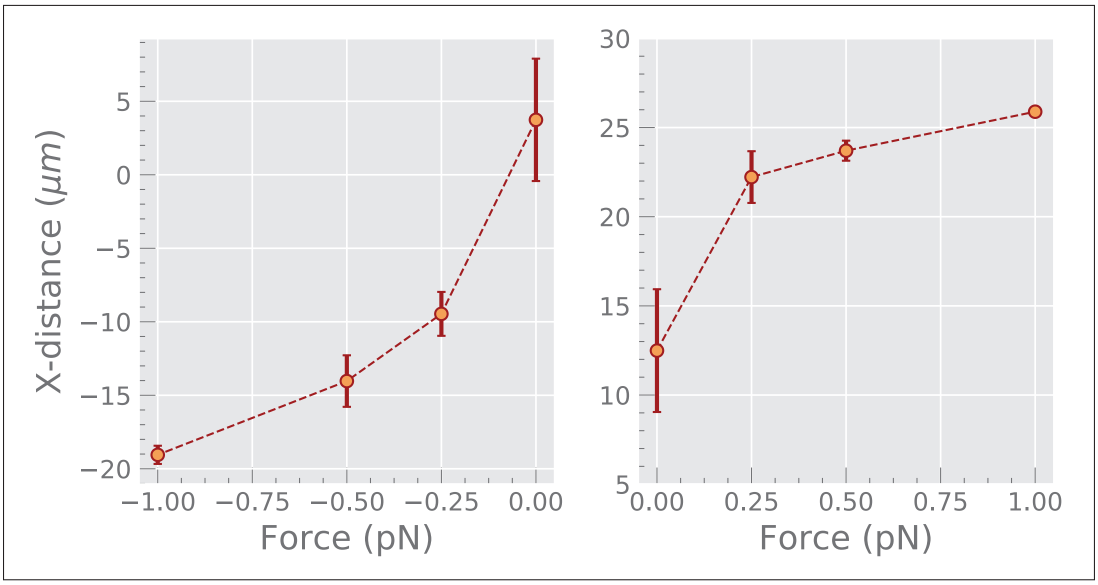


Figure 3.44: Effectiveness of applying an external force field. Left: negative force field, Right: positive force field

3.3.2 Effect on Speed

Figure 3.45 shows the speed analysis for the above simulation trajectories, compared to the results of an experiment [61]. The speed of the actin filament increases slightly in simulation, whereas in experiment, the increase in speed is more pronounced. This could be due to inadvertent heating of the substrate or the motility assay as the external electric field is applied.

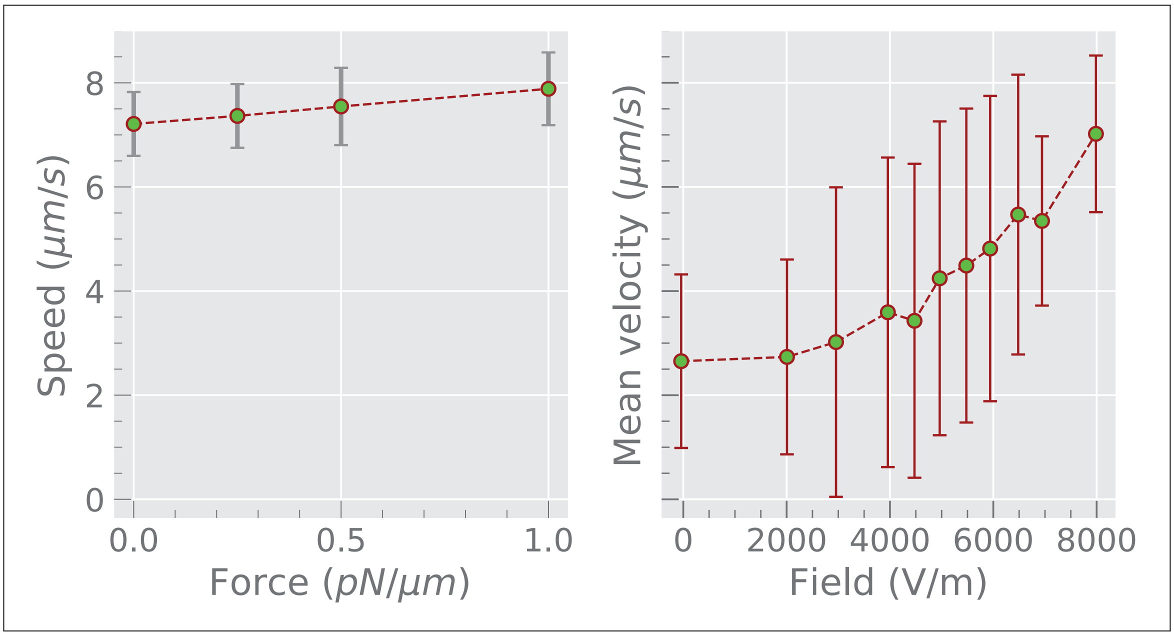


Figure 3.45: Actin filament speed under influence of an external force field. Left: simulation results, and Right: experimental results adopted from *Zalinge et. al.* [61].

3.3.3 Path Persistence Length

The L_p of the actin filament is calculated from the data of the leading tip trajectory. As explained in Section 2.4, different lengths of b_s can be used by choosing various Δt of the actin filament trajectory. As seen in Figure 3.46, at the beginning of the plot, a small Δt shows a correlation that is significantly different from the first correlation of 1. This is because a small Δt carries more information about the wiggling behaviour of the actin filament, reducing the chance that the trajectory steps are correlated, and hence a sudden drop in the correlation value. In this case, $\Delta t = 0.20$ *sec* was used. This value showed a smooth transition from the correlation of 1 to the next and was therefore a better predictor of the average value of the actin filament L_p.

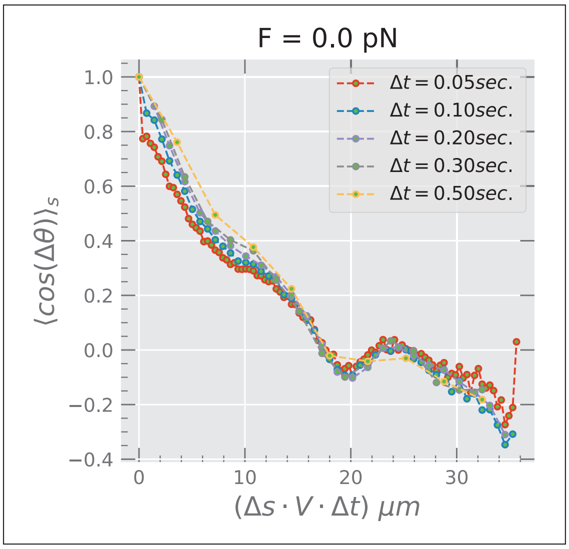


Figure 3.46: Path persistence changes with changing Δt. $\Delta t = 0.2s$ is used to calculate the L_p.

Figure 3.47 shows various trajectories of the actin filament when no external force field is applied. As explained in Figure 3.46, a lower Δt (blue line) is used during the calculation of L_p. Each actin filament follows a unique trajectory. The average movement and L_p of these trajectories without the influence of an external force are compared with other trajectories when an external force is applied.

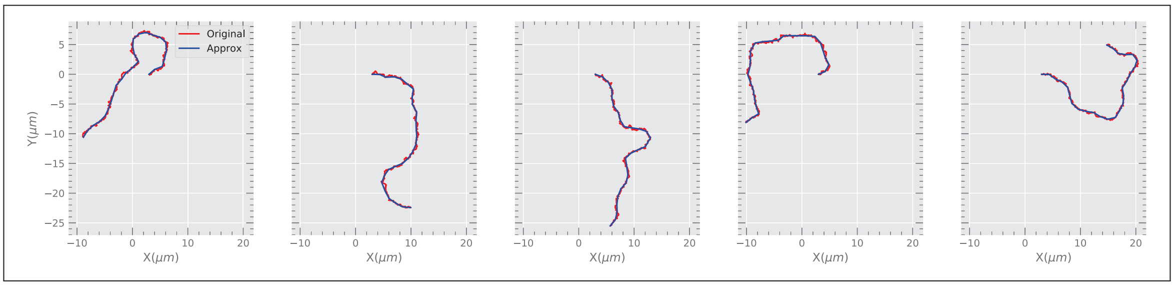


Figure 3.47: Sample trajectories when no external force is applied. The red line shows the original trajectory from the simulation data at $\Delta t = 0.01s$, and the blue line shows approximate trajectory at $\Delta t = 0.2s$

Figure 3.48 shows the results of the calculation of L_p as demonstrated by Algorithm 6 and implemented in Appendix B.1. The distance covered by the actin filament trajectories is plotted against the mean correlation of each step of the trajectories in Figure 3.47. In Figure 3.48(left), these correlations are shown by light blue dashed dots, and the mean of the five trajectories is shown by the red dashed dots. The L_p of the actin filament at this zero force is 3.7 ± 0.2 as shown in Figure 3.48(right). Some points are lost during logarithmic fitting due to negative correlations of the actin filament $(log(-x) = NaN)$.

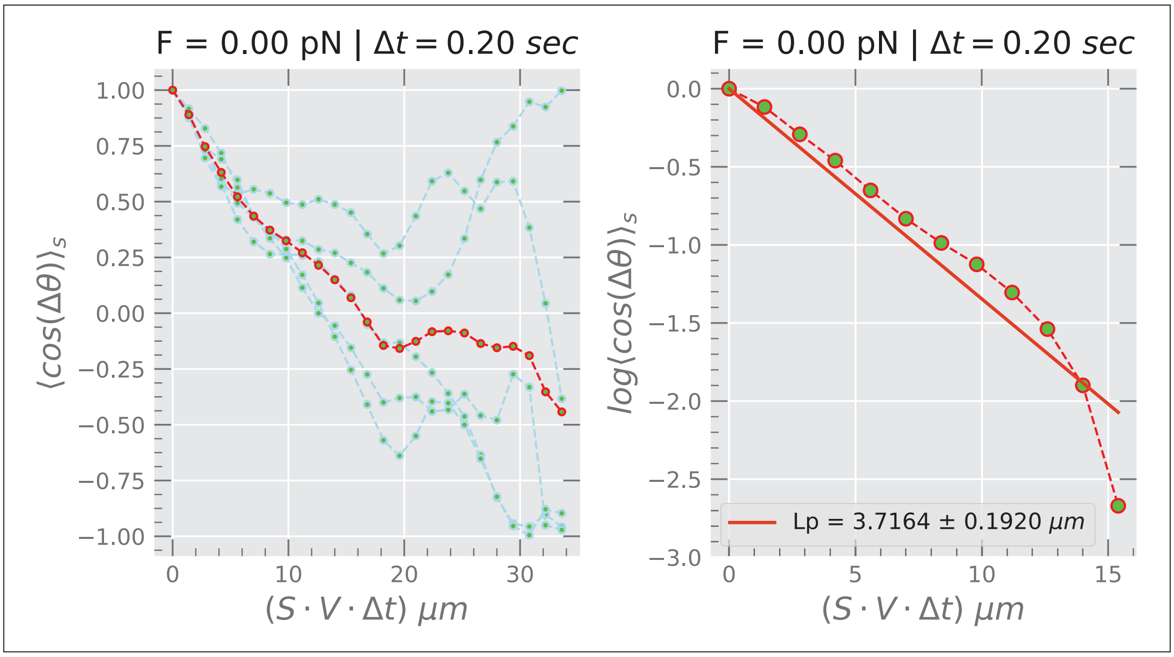


Figure 3.48: (Left): The mean correlation of each pair of steps of various trajectories when no external force is applied. (Right): L_p when no external force is applied

Calculating L_p of the actin filament as shown in Figure 3.48(left) is prone to errors.

In the process of calculating the correlations of the trajectory steps, the subjects of the trajectory steps (U_{bs}) from which to calculate the correlation drift far apart and the data points from which to compute the mean continue to decrease. Toward the end of the correlation plot, the data become sparse and therefore statistically insignificant and not sufficient for an accurate assessment of L_p. To mitigate this problem, around 10% of the correlation data is used to calculate the L_p of the actin filament. Figure 3.49 shows the calculation of L_p using only 30% of the data in Figure 3.48. The L_p of the actin filament improves slightly to 4.3 ± 0.1.

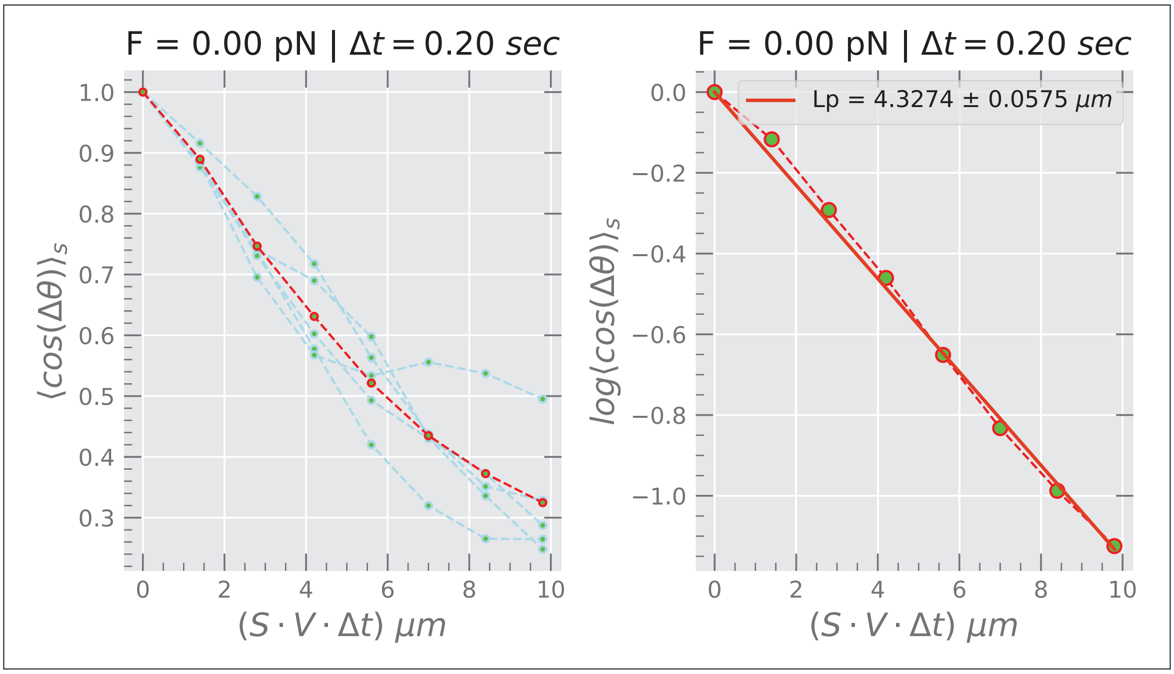


Figure 3.49: (Left): The mean correlation of each pair of steps of various trajectories when no external force is applied. (Right): L_p when no external force is applied

Figures 3.50 and 3.52 show the leading tip trajectories of various actin filaments when external forces of $+1.0pN$ and $-1.0pN$ are applied, respectively. Compared to the above trajectories, when no external force is applied, the application of an external force causes the actin filaments to travel in a rather straight path. As a result, calculating the correlations yields values that are close to 1 (≈ 0 after taking *log*), along the trajectory, and therefore the correlation values do not decay. The results shown in Figures 3.51 and 3.53 indicate a significant improvement in the actin filament L_p. However, this method may not be an appropriate measure for L_p of almost straight trajectories, since most correlations lie along the horizontal axis close to zero, compromising the fit. Other methods are recommended (*Jonathon Howard book, appendix section* [11]).

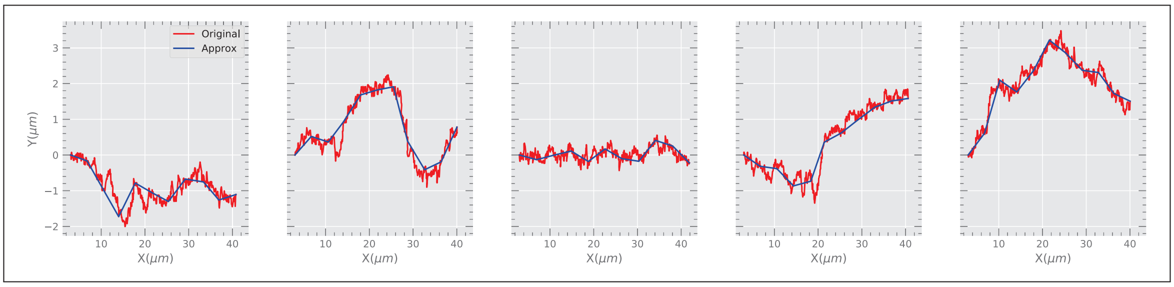


Figure 3.50: Sample trajectories when an external force of $+1.0pN$ is applied

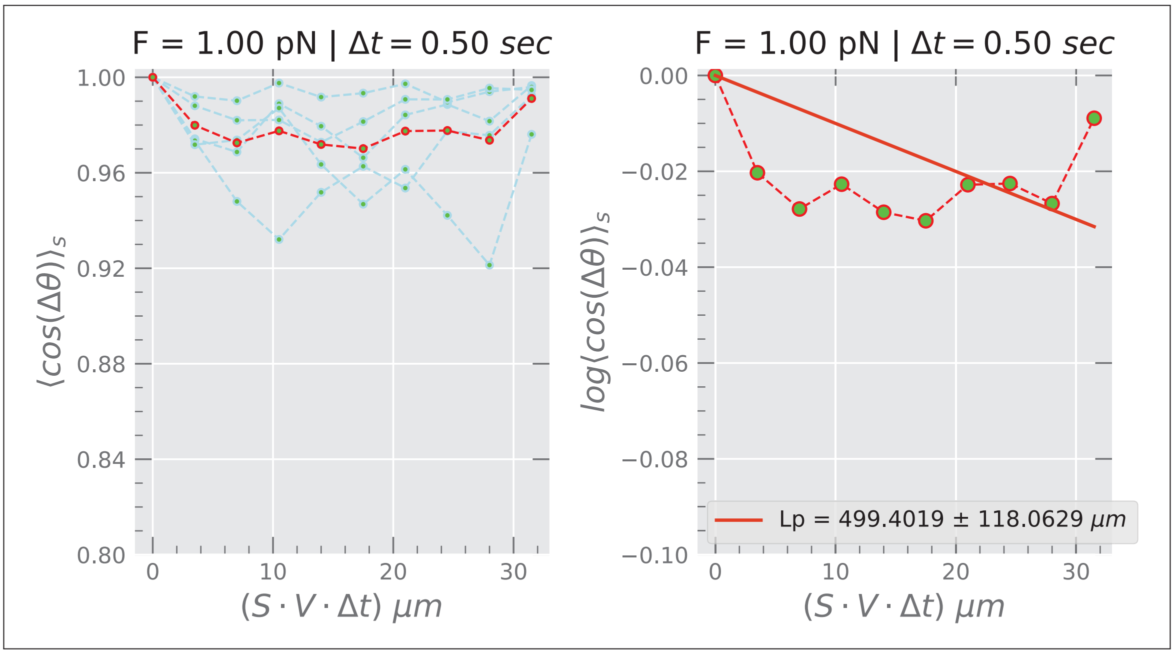


Figure 3.51: (Left): The mean correlation of each pair of steps of various trajectories when an external force of $+1pN$ is applied. (Right): L_p when no external force is applied

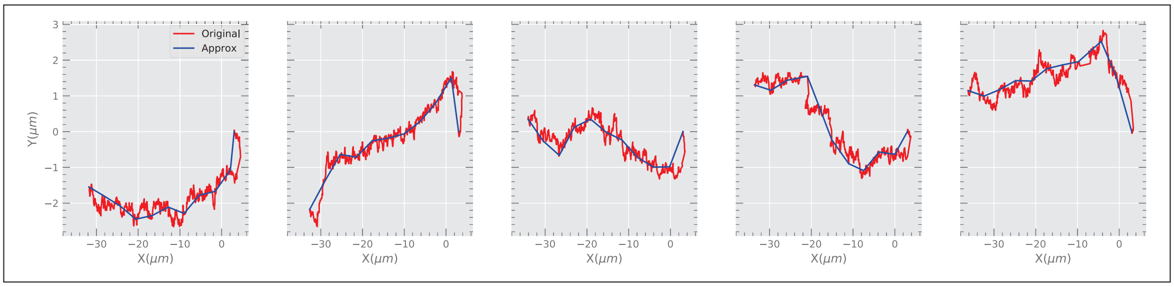


Figure 3.52: Sample trajectories when an external force of $-1.0pN$ is applied

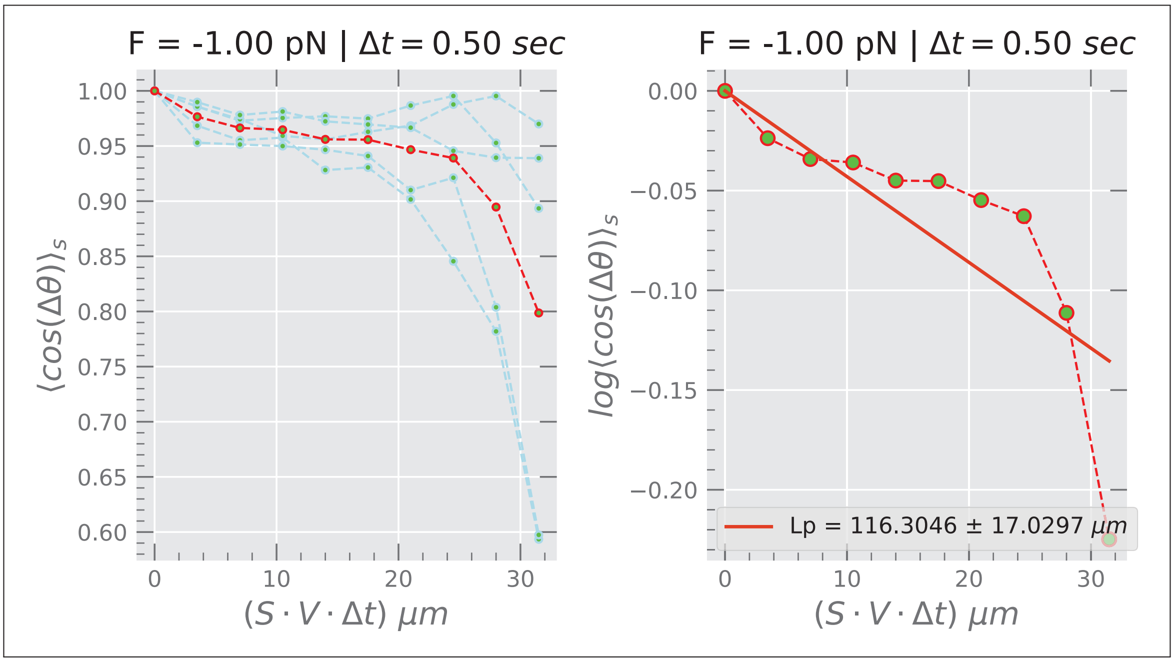


Figure 3.53: (Left): The mean correlation of each pair of steps of various trajectories when an external force of $-1pN$ is applied. (Right): L_p when no external force is applied

3.3.4 Biosensor Design Insight

Figure 3.54 shows a possible application for the use of an external force field as a possible drug delivery device. The ability to control actin filaments using an external force field allows the construction of designs similar to the microtubule channel-based rectifiers reported in other studies [82] [83]. To make an actin-based biomolecular rectifier, more sophisticated microfabricated channels are required, as shown in Figure 3.54(a). On the contrary, an equivalent biomolecular rectifier, possibly more efficient, shown in Figure 3.54(b) can be made by applying a negative force without having to incur the cost of complex or narrow microfabricated channels. Although Figure 3.54 (b) is less complex, the channel size does not have to be thin, as seen in Figure 3.42, an increase in the applied external force can be manipulated to adjust the width within which the filaments travel. The molecular shuttles may also travel longer and faster.

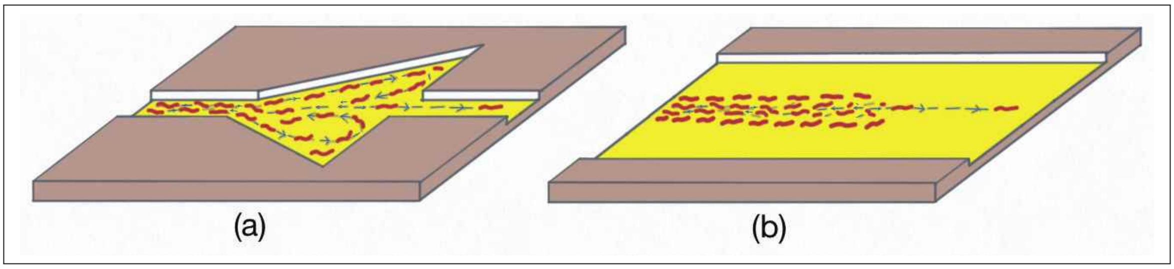


Figure 3.54: Biomolecular rectifier. (a) A rectifier made without the application of external force. (b) A rectifier made by making use of the applied force.

3.4 Recommendations

Gradient Surface

Figure 3.55 shows the result of the trajectories of simulated actin filaments moving in a motility assay with gradient myosin density. The placement of myosin in this simulation is such that the surface density imitates a gradient function. The AFs were observed to travel longer distances with a narrower channel. More research is needed to confirm these results.

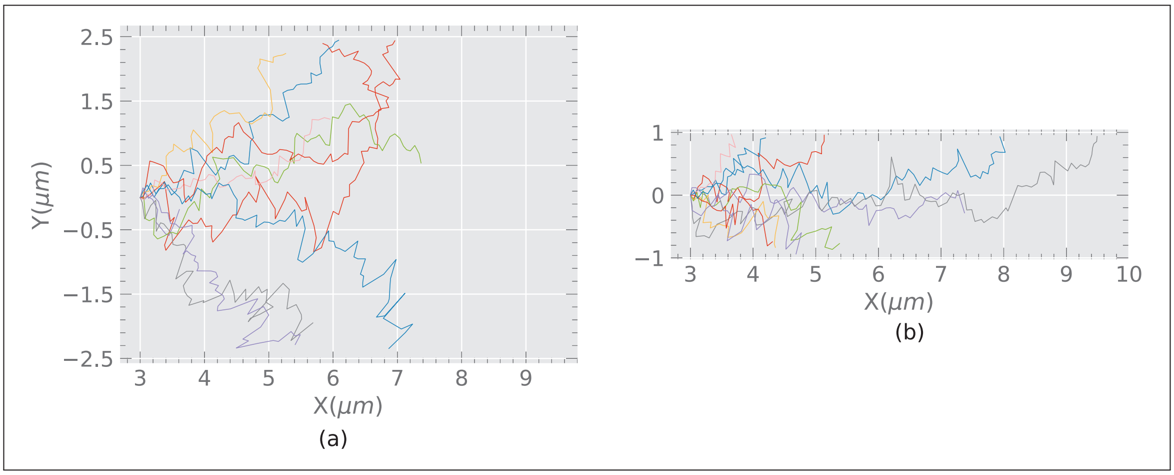


Figure 3.55: Gradient surface trajectories of a wide (a) versus a narrow channel (b)

CONCLUSION

"The critical fraction for the outset of the gliding movement of actin filaments can be elucidated by the difference in binding lifetimes of active and defective motors"

Biosensors and Bioelectronics
SAMUEL MACHARIA KANG'IRI *et al.*, 2022

4.1 Biosensor Design

For the actin filament to be propelled in an in vitro motility assay, a high percentage of active myosins on the substrate are required. Our 3D simulation shows that a motility assay with an active ratio greater than 90% is required to propel the actin filament. This was validated by our 1D analytical model which showed that a critical active ratio of 80% is required for progressive movement of the actin filament. The lower percentage of 80% maybe attributed to Brownian movements and thermal fluctuations that were not considered in the 1D analytical model. The high active ratio requirements remain regardless of length of the actin filament, ATP supplied, or the motor density of the motility assay. It can be deduced that the active ratio of the actomyosin motility assay is the single factor that influences the smooth gliding of the actin filaments. A biosensor integrated with actin and myosin maybe adversely affected by the presence of defective myosin by making it difficult for smooth gliding to be realized. These results emphasize on the importance of the cleaning process to remove the defective motors during the preparation of the in vitro motility assay.

When the actomyosin invitro motility assay is prepared, the substrate material should be carefully chosen to minimise myosin denature when adhered to the surface of the substrate. The hydrophobicity of the substrate material should be considered since a higher material hydrophobicity has been associated with low active myosin molecules. Minimising the population of myosins that become defective at the substrate level during the initial stage may have a significant contribution to sustaining active myosin in the in vitro motility assay.

The active myosin binding lifetime is a function of binding defective myosin. A motility assay of between 70% and 90% active myosin has the highest number of active myosins binding to the actin filament. In addition, these myosins spend the longest time bound to the actin filament. It can be conjectured that binding active myosin increases to try and overcome the resisting force of the defective myosins. When the active myosin ratio is increased to around 92%, gliding is commenced. This implies that even when the in vitro motility assay has a few defective myosins, the greater population of active myosins can still propel the actin filament.

4.2 Translation Mechanism

The spiral shape formed by the actin filament may be attributed to the presence of defective myosin. In our simulation of the 3D motility assay, we observed that the actin filament was prone to curving during the transition phase. Because the actin filament is semi-flexible, it is easier to move it all by pulling rather than pushing. This implies that if the movement of the actin filament is abruptly impeded at the leading tip by a defective myosin, the trailing end may continue to move for a while, making a curve around the impeding fulcrum. For applications that involve the movement of cargo through channels, the tendency to form spiral shapes is a defect that reduces the persistence length of the actin filament. Minimising the denaturation of the adhering myosins on the substrate surface should minimise these spiral shapes. Alternatively, using an actin filament bundle would stiffen the actin filament, thus minimizing the possibility of forming spiral shapes.

The alignment of the actin filament is significantly improved by the application of an electric force field. In our simulation of applying an external force to the actin filaments, we were able to reproduce in silico the movement of the actin filaments when they were subjected to an electric field. This enables us to conduct several tests about the electric motility and build up intuition based on the analysed results. These insights are important for the design and testing of actin-based devices guided by electric field.

REFERENCES

[1] H. L. Sweeney and D. W. Hammers, "Muscle contraction," *Cold Spring Harbor Perspectives in Biology*, vol. 10, no. 2, Feb. 2018, ISSN: 19430264. DOI: 10.1101/cshperspect.a023200.

[2] R. M. N. Alexander, *Principles of animal locomotion*, 3. Jun. 2013, vol. 85, pp. 1–371, ISBN: 9781400849512. DOI: 10.1644/1545-1542(2004)085<0584:br>2.0.co;2.

[3] N. C. Sharp, "Timed running speed of a cheetah (Acinonyx jubatus)," *Journal of Zoology*, vol. 241, no. 3, pp. 493–494, Mar. 1997, ISSN: 09528369. DOI: 10.1111/j.1469-7998.1997.tb04840.x.

[4] C. N. Joyce Chen, L. D. Thompson, and L. A. Snow, "Muscle Structure and Function," in *Orthopaedic Physical Therapy Secrets: Third Edition*, Elsevier, Jan. 2017, ch. 1, pp. 1–9, ISBN: 9780323286855. DOI: 10.1016/B978-0-323-28683-1.00001-1.

[5] J. M. Squire, "Muscle contraction: Sliding filament history, sarcomere dynamics and the two Huxleys," *Global Cardiology Science and Practice*, vol. 2016, no. 2, Oct. 2016, ISSN: 2305-7823. DOI: 10.21542/gcsp.2016.11.

[6] S. J. Kron and J. A. Spudich, "Fluorescent actin filaments move on myosin fixed to a glass surface," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 83, no. 17, pp. 6272–6276, 1986, ISSN: 00278424. DOI: 10.1073/pnas.83.17.6272.

[7] L. M. Scheid, C. Weber, N. Bopp, M. Mosqueira, and R. H. Fink, "Extraction protocols for individual zebrafish's ventricle myosin and skeletal muscle actin for in vitro motility assays," *Frontiers in Physiology*, vol. 8, no. MAY, p. 367, May 2017, ISSN: 1664042X. DOI: 10.3389/fphys.2017.00367.

[8] C. Toepfer and J. R. Sellers, *Use of fluorescent techniques to study the in vitro movement of myosins.* 2014. DOI: 10.1007/978-3-0348-0856-9{_}9.

[9] M. U. H. C. Dr. Anne-Marie Lauzon's Laboratory, *Fluorescently labeled actin filaments propelled by smooth muscle myosin purified from chicken gizzard.* 2014.

[10] K. Mukund and S. Subramaniam, *Skeletal muscle: A review of molecular structure and function, in health and disease*, Jan. 2020. DOI: 10.1002/wsbm.1462.

[11] J. Howard, *Mechanics of motor proteins and the cytoskeleton*. Sinauer Associates, Publishers, Mar. 2001, p. 384, ISBN: 978-0878933334. DOI: 10.1063/1.1472396.

[12] M. A. Goldstein and J. Cartwright, "Microtubules in Adult Mammalian Muscle," in *Cell and Muscle Motility*, Boston, MA: Springer US, 1982, pp. 85–92. DOI: 10.1007/978-1-4684-4037-9{_}7.

[13] G. Saper and H. Hess, "Synthetic Systems Powered by Biological Molecular Motors.," *Chemical reviews*, vol. 120, no. 1, pp. 288–309, Jan. 2020, ISSN: 1520-6890. DOI: 10.1021/acs.chemrev.9b00249.

[14] G. D. Bachand, N. F. Bouxsein, V. VanDelinder, and M. Bachand, "Biomolecular motors in nanoscale materials, devices, and systems," *WIREs Nanomedicine and Nanobiotechnology*, vol. 6, no. 2, pp. 163–177, Mar. 2014, ISSN: 1939-5116. DOI: 10.1002/wnan.1252.

[15] A. Agarwal and H. Hess, "Biomolecular motors at the intersection of nanotechnology and polymer science," *Progress in Polymer Science*, vol. 35, no. 1-2, pp. 252–277, Jan. 2010, ISSN: 00796700. DOI: 10.1016/j.progpolymsci.2009.10.007.

[16] D. J. G. Bakewell and D. V. Nicolau, "Protein Linear Molecular Motor-Powered Nanodevices," *Australian Journal of Chemistry*, vol. 60, no. 5, pp. 314–332, May 2007, ISSN: 0004-9425. DOI: 10.1071/CH06456.

[17] T. Korten, A. Månsson, and S. Diez, *Towards the application of cytoskeletal motor proteins in molecular detection and diagnostic devices*, Aug. 2010. DOI: 10.1016/j.copbio.2010.05.001.

[18] A. Månsson, *Translational actomyosin research: Fundamental insights and applications hand in hand*, May 2012. DOI: 10.1007/s10974-012-9298-5.

[19] C. Dong and C. Z. Dinu, *Molecular trucks and complementary tracks for bionanotechnological applications*, Aug. 2013. DOI: 10.1016/j.copbio.2013.01.007.

[20] G. D. Bachand, H. Hess, B. Ratna, P. Satir, and V. Vogel, "SMART DUST BIOSENSORS POWERED BY BIOMOLECULAR MOTORS," *Lab on a Chip*, vol. 9, no. 12, pp. 1661–1666, 2009, ISSN: 1473-0197. DOI: 10.1039/b821055a.

[21] C. T. Lin, M. T. Kao, K. Kurabayashi, and E. Meyhofer, "Self-contained, biomolecular motor-driven protein sorting and concentrating in an ultrasensitive microfluidic chip," *Nano Letters*, vol. 8, no. 4, pp. 1041–1046, 2008, ISSN: 15306984. DOI: 10.1021/nl072742x.

[22] T. Fischer, A. Agarwal, and H. Hess, "A smart dust biosensor powered by kinesin motors," *Nature Nanotechnology*, vol. 4, no. 3, pp. 162–166, Mar. 2009, ISSN: 17483395. DOI: 10.1038/nnano.2008.393.

[23] M. Lard, L. ten Siethoff, S. Kumar, M. Persson, G. te Kronnie, H. Linke, and A. Månsson, "Ultrafast molecular motor driven nanoseparation and biosensing," *Biosensors and Bioelectronics*, vol. 48, pp. 145–152, Oct. 2013, ISSN: 09565663. DOI: 10.1016/j.bios.2013.03.071.

[24] N. Isozaki, H. Shintaku, H. Kotera, T. L. Hawkins, J. L. Ross, and R. Yokokawa, "Control of molecular shuttles by designing electrical and mechanical properties of microtubules," *Science Robotics*, vol. 2, no. 10, eaan4882, Sep. 2017, ISSN: 24709476. DOI: 10.1126/scirobotics.aan4882.

[25] D. V. Nicolau, M. Lard, T. Korten, F. C. Van Delft, M. Persson, E. Bengtsson, A. Månsson, S. Diez, H. Linke, and D. V. Nicolau, "Parallel computation with molecular-motor-propelled agents in nanofabricated networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 113, no. 10, pp. 2591–2596, 2016, ISSN: 10916490. DOI: 10.1073/pnas.1510825113.

[26] T. Nakano, M. J. Moore, F. Wei, A. V. Vasilakos, and J. Shuai, "Molecular communication and networking: opportunities and challenges.," *IEEE transactions on nanobioscience*, vol. 11, no. 2, pp. 135–48, Jun. 2012, ISSN: 1558-2639. DOI: 10.1109/TNB.2012.2191570.

[27] N. Farsad, H. B. Yilmaz, A. Eckford, C.-B. Chae, and W. Guo, "A Comprehensive Survey of Recent Advancements in Molecular Communication," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 3, pp. 1887–1919, Oct. 2014, ISSN: 1553877X. DOI: 10.1109/COMST.2016.2527741.

[28] H. Hess, *Self-assembly driven by molecular motors*, Jul. 2006. DOI: 10.1039/b518281f.

[29] T. Nitta, Y. Wang, Z. Du, K. Morishima, and Y. Hiratsuka, "A printable active network actuator built from an engineered biomolecular motor," *Nature Materials*, pp. 1–7, Apr. 2021, ISSN: 14764660. DOI: 10.1038/s41563-021-00969-6.

[30] H. Hess and G. D. Bachand, "Biomolecular motors," *Materials Today*, vol. 8, no. 12, pp. 22–29, Dec. 2005, ISSN: 1369-7021. DOI: 10.1016/S1369-7021(05)71286-4.

[31] R. Eelkema, M. M. Pollard, J. Vicario, N. Katsonis, B. S. Ramon, C. W. Bastiaansen, D. J. Broer, and B. L. Feringa, "Nanomotor rotates microscale objects," *Nature*, vol. 440, no. 7081, p. 163, Mar. 2006, ISSN: 14764687. DOI: 10.1038/440163a.

[32] B. L. Feringa, *The Art of Building Small: From Molecular Switches to Motors (Nobel Lecture)*, Sep. 2017. DOI: 10.1002/anie.201702979.

[33] M. Samandari, M. G. Julia, A. Rice, A. Chronopoulos, and A. E. del Rio Hernandez, *Liquid biopsies for management of pancreatic cancer*, Nov. 2018. DOI: 10.1016/j.trsl.2018.07.008.

[34] M. Nehra, M. Lettieri, N. Dilbaghi, S. Kumar, and G. Marrazza, "Nano-Biosensing Platforms for Detection of Cow's Milk Allergens: An Overview," *Sensors 2020, Vol. 20, Page 32*, vol. 20, no. 1, p. 32, Dec. 2019. DOI: 10.3390/S20010032.

[35] M. A. Lifson, M. O. Ozen, F. Inci, S. Q. Wang, H. Inan, M. Baday, T. J. Henrich, and U. Demirci, *Advances in biosensing strategies for HIV-1 detection, diagnosis, and therapeutic monitoring*, Aug. 2016. DOI: 10.1016/j.addr.2016.05.018.

[36] S. Senapati, A. K. Mahanta, S. Kumar, and P. Maiti, *Controlled drug delivery vehicles for cancer treatment and their performance*, Dec. 2018. DOI: 10.1038/s41392-017-0004-3.

[37] T. A. Slastnikova, A. V. Ulasov, A. A. Rosenkranz, and A. S. Sobolev, "Targeted intracellular delivery of antibodies: The state of the art," *Frontiers in Pharmacology*, vol. 9, no. OCT, p. 1208, Oct. 2018, ISSN: 16639812. DOI: 10.3389/fphar.2018.01208.

[38] L. G. Lee, E. S. Nordman, M. D. Johnson, and M. F. Oldham, "A Low-Cost, High-Performance System for Fluorescence Lateral Flow Assays," *Biosensors 2013, Vol. 3, Pages 360-373*, vol. 3, no. 4, pp. 360–373, Oct. 2013. DOI: 10.3390/BIOS3040360.

[39] L. Clime, D. Brassard, M. Geissler, and T. Veres, "Active pneumatic control of centrifugal microfluidic flows for lab-on-a-chip applications," *Lab on a Chip*, vol. 15, no. 11, pp. 2400–2411, 2015, ISSN: 14730189. DOI: 10.1039/c4lc01490a.

[40] J. Chen, Y. Yu, S. Yang, S. K. Fan, and J. Zhou, "A monolithic lab-on-a-chip for electrochemical detection," in *ICSICT 2012 - 2012 IEEE 11th International Conference on Solid-State and Integrated Circuit Technology, Proceedings*, IEEE, Oct. 2012, pp. 1–3, ISBN: 9781467324724. DOI: 10.1109/ICSICT.2012.6467957.

[41] D. Leech, *Affinity biosensors*, Jan. 1994. DOI: 10.1039/CS9942300205.

[42] K. Kochanowski, B. Volkmer, L. Gerosa, B. R. Van Rijsewijk, A. Schmidt, and M. Heinemann, "Functioning of a metabolic flux sensor in Escherichia coli," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 110, no. 3, pp. 1130–1135, Jan. 2013, ISSN: 10916490. DOI: 10.1073/pnas.1202582110.

[43] E. Brauns, E. Morsbach, M. Baumer, and W. Lang, "A fast and sensitive catalytic hydrogen sensor based on a stabilized nanoparticle catalyst," *2013 Transducers and Eurosensors XXVII: The 17th International Conference on Solid-State Sensors, Actuators and Microsystems, TRANSDUCERS and EUROSENSORS 2013*, pp. 1178–1181, 2013. DOI: 10.1109/TRANSDUCERS.2013.6626983.

[44] C. Karunakaran, K. Bhargava, and R. Benjamin, *Biosensors and Bioelectronics*. Elsevier, Jul. 2015, pp. 1–320, ISBN: 9780128031018. DOI: 10.1016/C2014-0-03790-2.

[45] M. Lard, L. t. Siethoff, A. Månsson, and H. Linke, "Tracking actomyosin at fluorescence check points," *Scientific Reports*, vol. 3, no. 1, pp. 1–7, Jan. 2013, ISSN: 20452322. DOI: 10.1038/srep01092.

[46] R. S. Malon, S. Sadir, M. Balakrishnan, and E. P. Córcoles, *Saliva-Based Biosensors: Noninvasive Monitoring Tool for Clinical Diagnostics*, 2014. DOI: 10.1155/2014/962903.

[47] J. Kerssemakers, L. Ionov, U. Queitsch, S. Luna, H. Hess, and S. Diez, "3D nanometer tracking of motile microtubules on reflective surfaces," *Small*, vol. 5, no. 15, pp. 1732–1737, 2009, ISSN: 16136810. DOI: 10.1002/smll.200801388.

[48] M. A. Rahman, A. Salhotra, and A. Månsson, "Comparative analysis of widely used methods to remove nonfunctional myosin heads for the in vitro motility assay," *Journal of Muscle Research and Cell Motility*, vol. 39, no. 5-6, pp. 175–187, Dec. 2018, ISSN: 15732657. DOI: 10.1007/s10974-019-09505-1.

[49] F. Gittes, B. Mickey, J. Nettleton, and J. Howard, "Flexural rigidity of microtubules and actin filaments measured from thermal fluctuations in shape," *Journal of Cell Biology*, vol. 120, no. 4, pp. 923–934, Feb. 1993, ISSN: 00219525. DOI: 10.1083/jcb.120.4.923.

[50] K. Fujita, M. Ohmachi, K. Ikezaki, T. Yanagida, and M. Iwaki, "Direct visualization of human myosin II force generation using DNA origami-based thick filaments," *Communications Biology*, vol. 2, no. 1, pp. 1–11, Dec. 2019, ISSN: 23993642. DOI: 10.1038/s42003-019-0683-0.

[51] T. C. Campbell and H. Jacobson, "Whole: rethinking the science of nutrition," *Choice Reviews Online*, vol. 51, no. 02, pp. 51–0916, Oct. 2013, ISSN: 0009-4978. DOI: 10.5860/choice.51-0916.

[52] Y. Ishigure and T. Nitta, "Simulating an Actomyosin in Vitro Motility Assay: Toward the Rational Design of Actomyosin-Based Microtransporters," *IEEE Transactions on Nanobioscience*, vol. 14, no. 6, pp. 641–648, Sep. 2015, ISSN: 15361241. DOI: 10.1109/TNB.2015.2443373.

[53] S. Walcott, D. M. Warshaw, and E. P. Debold, "Mechanical coupling between myosin molecules causes differences between ensemble and single-molecule measurements," *Biophysical Journal*, vol. 103, no. 3, pp. 501–510, 2012, ISSN: 00063495. DOI: 10.1016/j.bpj.2012.06.031.

[54] T. Nishizaka, H. Miyata, H. Yoshikawa, S. Ishiwata, and K. Kinosita, "Unbinding force of a single motor molecule of muscle measured using optical tweezers.," *Nature*, vol. 377, no. 6546, pp. 251–4, Sep. 1995, ISSN: 0028-0836. DOI: 10.1038/377251a0.

[55] U. M. Ascher, S. J. Ruuth, and R. J. Spiteri, "Implicit-Explicit Runge-Kutta Methods for Time-Dependent Partial Diierential Equations," *APPL. NUMER. MATH*, vol. 25, pp. 151–167, 1997.

[56] F. Nedelec and D. Foethke, "Collective Langevin dynamics of flexible cytoskeletal fibers," *New Journal of Physics*, 2007, ISSN: 13672630. DOI: 10.1088/1367-2630/9/11/427.

[57] Y. Ishigure and T. Nitta, "Understanding the Guiding of Kinesin/Microtubule-Based Microtransporters in Microfabricated Tracks," *Langmuir*, 2014, ISSN: 15205827. DOI: 10.1021/la5021884.

[58] T. Q. Uyeda, S. J. Kron, and J. A. Spudich, "Myosin step size. Estimation from slow sliding movement of actin over low densities of heavy meromyosin," *Journal of Molecular Biology*, vol. 214, no. 3, pp. 699–710, 1990, ISSN: 00222836. DOI: 10.1016/0022-2836(90)90287-V.

[59] T. Duke, T. E. Holy, and S. Leibler, “"Gliding assays" for motor proteins: A theoretical analysis,” *Physical Review Letters*, vol. 74, no. 2, pp. 330–333, Jan. 1995, ISSN: 00319007. DOI: 10.1103/PhysRevLett.74.330.

[60] G. E. P. Box and M. E. Muller, “A Note on the Generation of Random Normal Deviates,” *The Annals of Mathematical Statistics*, vol. 29, no. 2, pp. 610–611, Jun. 1958, ISSN: 0003-4851. DOI: 10.1214/aoms/1177706645.

[61] H. Van Zalinge, L. C. Ramsey, J. Aveyard, M. Persson, A. Mansson, and D. V. Nicolau, “Surface-Controlled Properties of Myosin Studied by Electric Field Modulation,” *Langmuir*, vol. 31, no. 30, pp. 8354–8361, 2015, ISSN: 15205827. DOI: 10.1021/acs.langmuir.5b01549.

[62] D. Riveline, A. Ott, F. Jülicher, D. A. Winkelmann, O. Cardoso, J.-J. Lacapère, S. Magnúsdóttir, J. L. Viovy, L. Gorre-Talini, and J. Prost, “Acting on actin: the electric motility assay,” *European Biophysics Journal*, vol. 27, no. 4, pp. 403–408, Jun. 1998, ISSN: 0175-7571. DOI: 10.1007/s002490050147.

[63] S. M. Kang’iri, A. Salem, D. V. Nicolau, and T. Nitta, “Effects of defective motors on the active transport in biosensors powered by biomolecular motors,” *Biosensors and Bioelectronics*, vol. 203, p. 114 011, May 2022, ISSN: 09565663. DOI: 10.1016/j.bios.2022.114011.

[64] L. Scharrel, R. Ma, R. Schneider, F. Jülicher, and S. Diez, “Multimotor transport in a system of active and inactive kinesin-1 motors,” *Biophysical Journal*, 2014, ISSN: 15420086. DOI: 10.1016/j.bpj.2014.06.014.

[65] L. Bourdieu, T. Duke, M. B. Elowitz, D. A. Winkelmann, S. Leibler, and A. Libchaber, “Spiral defects in motility assays: A measure of motor protein force,” *Physical Review Letters*, vol. 75, no. 1, pp. 176–179, 1995, ISSN: 00319007. DOI: 10.1103/PhysRevLett.75.176.

[66] G. H. Koenderink and E. K. Paluch, *Architecture shapes contractility in actomyosin networks*, Feb. 2018. DOI: 10.1016/j.ceb.2018.01.015.

[67] M. Murrell, P. W. Oakes, M. Lenz, and M. L. Gardel, *Forcing cells into shape: The mechanics of actomyosin contractility*, Jul. 2015. DOI: 10.1038/nrm4012.

[68] V. P. Shirinsky, K. G. Biryukov, J. M. Hettasch, and J. R. Sellers, “Inhibition of the relative movement of actin and myosin by caldesmon and calponin,” *Journal of Biological Chemistry*, vol. 267, no. 22, pp. 15 886–15 892, 1992, ISSN: 00219258. DOI: 10.1016/S0021-9258(19)49617-8.

[69] S. J. Kron, Y. Y. Toyoshima, T. Q. Uyeda, and J. A. Spudich, “Assays for Actin Sliding Movement over Myosin-Coated Surfaces,” *Methods in Enzymology*, vol. 196, no. C, pp. 399–416, Jan. 1991, ISSN: 15577988. DOI: 10.1016/0076-6879(91)96035-P.

[70] T. Nitta, A. Tanahashi, M. Hirano, and H. Hess, “Simulating molecular shuttle movements: Towards computer-aided design of nanoscale transport systems,” *Lab on a Chip*, vol. 6, no. 7, pp. 881–885, 2006, ISSN: 14730189. DOI: 10.1039/b601754a.

[71] Y. S. Cheng, F. De Souza Leite, and D. E. Rassier, "The load dependence and the force-velocity relation in intact myosin filaments from skeletal and smooth muscles," *American Journal of Physiology - Cell Physiology*, vol. 318, no. 1, pp. C103–C110, Jan. 2020, ISSN: 15221563. DOI: 10.1152/ajpcell.00339.2019.

[72] C. Veigel, J. E. Molloy, S. Schmitz, and J. Kendrick-Jones, "Load-dependent kinetics of force production by smooth muscle myosin measured with optical tweezers," *NATURE CELL BIOLOGY*, vol. 5, 2003. DOI: 10.1038/ncb1060.

[73] Q. W. Shen and D. R. Swartz, "Influence of salt and pyrophosphate on bovine fast and slow myosin S1 dissociation from actin," *Meat science*, vol. 84, no. 3, p. 364, Mar. 2010. DOI: 10.1016/J.MEATSCI.2009.09.003.

[74] K. L. Hanson, F. Fulga, S. Dobroiu, G. Solana, O. Kaspar, V. Tokarova, and D. V. Nicolau, "Polymer surface properties control the function of heavy meromyosin in dynamic nanodevices," *Biosensors and Bioelectronics*, vol. 93, pp. 305–314, Jul. 2017, ISSN: 09565663. DOI: 10.1016/j.bios.2016.08.061.

[75] R. Rossi, M. Maffei, R. Bottinelli, and M. Canepari, "Temperature dependence of speed of actin filaments propelled by slow and fast skeletal myosin isoforms," *Journal of Applied Physiology*, vol. 99, no. 6, pp. 2239–2245, 2005, ISSN: 87507587. DOI: 10.1152/japplphysiol.00543.2005.

[76] D. A. Winkelmann, L. Bourdieu, A. Ott, F. Kinose, and A. Libchaber, "Flexibility of myosin attachment to surfaces influences F-actin motion," *Biophysical Journal*, vol. 68, no. 6, pp. 2444–2453, 1995, ISSN: 00063495. DOI: 10.1016/S0006-3495(95)80426-1.

[77] R. W. Lymn and E. W. Taylor, "Mechanism of Adenosine Triphosphate Hydrolysis by Actomyosin," *Biochemistry*, vol. 10, no. 25, pp. 4617–4624, 1971, ISSN: 15204995. DOI: 10.1021/bi00801a004.

[78] A. M. Lauzon, M. J. Tyska, A. S. Rovner, Y. Freyzon, D. M. Warshaw, and K. M. Trybus, "A 7-amino-acid insert in the heavy chain nucleotide binding loop alters the kinetics of smooth muscle myosin in the laser trap," *Journal of Muscle Research and Cell Motility*, vol. 19, no. 8, pp. 825–837, 1998, ISSN: 01424319. DOI: 10.1023/A:1005489501357.

[79] N. M. Kad, J. B. Patlak, P. M. Fagnant, K. M. Trybus, and D. M. Warshaw, "Mutation of a conserved glycine in the SH1-SH2 helix affects the load-dependent kinetics of myosin," *Biophysical Journal*, vol. 92, no. 5, pp. 1623–1631, 2007, ISSN: 00063495. DOI: 10.1529/biophysj.106.097618.

[80] W. O. Fenn, "A quantitative comparison between the energy liberated and the work performed by the isolated sartorius muscle of the frog," *The Journal of Physiology*, vol. 58, no. 2-3, pp. 175–203, Dec. 1923, ISSN: 14697793. DOI: 10.1113/jphysiol.1923.sp002115.

[81] S. M. Kang'iri and T. Nitta, "A Mathematical Model Predicting Gliding Speed of Actin Molecular Shuttles Over Myosin Motors in the Presence of Defective Motors," in *Bio-Inspired Information and Communications Technologies. BICT 2021.* Ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, T. Nakano, Ed., vol. 403, Cham: Springer International Publishing, 2021, pp. 207–214, ISBN: 978-3-030-92162-0. DOI: 10.1007/978-3-030-92163-7{_}17.

[82] Y. Hiratsuka, T. Tada, K. Oiwa, T. Kanayama, and T. Q. Uyeda, "Controlling the direction of kinesin-driven microtubule movements along microlithographic tracks," *Biophysical Journal*, vol. 81, no. 3, pp. 1555–1561, Sep. 2001, ISSN: 00063495. DOI: 10.1016/S0006-3495(01)75809-2.

[83] M. G. Van Den Heuvel, C. T. Butcher, R. M. Smeets, S. Diez, and C. Dekker, "High rectifying efficiencies of microtubule motility on Kinesin-coated gold nanostructures," *Nano Letters*, vol. 5, no. 6, pp. 1117–1122, Jun. 2005, ISSN: 15306984. DOI: 10.1021/nl0506554.

APPENDIX

> "I came here confused about actin and myosin. Now I am still confused, but at a higher level."
>
> *Frontiers in Molecular Motors*
> Sir Andrew Huxley, 2000

To ensure uninterrupted dissertation flow, the following appendices show important parts of this study that maybe referenced in the main chapters but could not be placed in the main text.

A Simulation Program

A.1 Parameters Used

Table 1: Table of variables and parameters used in the simulation program

Parameter	Particulars	Value
NumAssay	Number of assays	1
NumFilament	Number of filaments used	1
NumBeads	Number of beads	13
NumType	Types of myosins	2
MaxNumIteration	Maximum number of iterations	1E6
AreaRenewDiv	The new area divisions	2E4
OutPutDiv	Steps for every output	2E4
MaxInteractinNum	Maximum interacting filaments	1
seed	Random seed	any integer value
NumTimeStep	Number of simulation time steps	6E6
TimeStepEquil	Time step equlibrium	1E9
MaxNumMotors	Maximum number of myosins	1E6
TimeForceON	Time for force action	1E9
pi	π value used	3.14159265358979
kBT	Energy used	0.0042 $pN.\mu m$
dt	Change in time used	5.0E-7
Tol	Tolerance allowed	1.0E-6
BondLength	Actin bond length	0.25
EI	Actin flexural rigidity	0.073 $pN.\mu m^2$ [49]
DiameterAF	Actin filament diameter	0.006
Motor_Density	Myosin density	3000.0
k	Myosin spring constant	300.0 $pN/\mu m$
Stepsize	Myosin step size	1.0E-2 μm
k_a	The myosin binding rate	40.0 s^{-1}
k_d0	No force rate constant	350.0 s^{-1}
k_t	Rate	2.0 μm^-1s^{-1}
ATP	Adenosine triphosphate	2000.0 μm
k_hp	Forward rate constant	100.0 s^{-1}
k_hm	Backward rate constant	10.0 s^{-1}
delta_x	Binding state-energy barrier distance	-1.86E-3
F_Motor_Detach1	Detachment force for motor type 1	9.2 pN
F_Motor_Detach2	Detachment force for motor type 2	9.2 pN
CaptureRadius	The myosin-actin capture radius	0.020 μm
Type1Ratio	Active myosin ratio	0.60
ExtForceDensity0	External force exerted	0.0 pN/um
ExtForceMag	Magnitude of the external force	0.0
XLimit	X-axis limit of the track surface	500.0 μm
OutPutDiv2	Alternative steps for every output	$NINT(0.0005/dt)$

A.2 Main Program Implementation

```
PROGRAM MAIN
 USE PARAMETERS
 USE mtmod !Random number generator
 USE FUNC
 USE OUTPUT
 USE PLANAR_TRACK_CONFINEMENT
 USE UNIFORM_FORCE_X
 IMPLICIT NONE
 INTEGER :: seed_Assay, I_Assay, I, J, AreaCounter, EraseCounter, &
 NumIteration, OutputfileCounter, UnitNumConformation, UnitNumTipXY, &
 UnitNumMotorStates, UnitNumAreaEraseCounter
 INTEGER(KIND = Range15):: TS, IM, ActiveMotorIdxOffset,&
 ActiveMotorIdxEnd
 INTEGER(KIND = Range15), DIMENSION(MaxAreaNum) :: AddedMotorNum
 INTEGER, DIMENSION(MaxNumMotors) ::   Release_ADP, MotorType
 !--- 2Motors: 1=Myosin, 2=Defective Myosin ---
 REAL(KIND = DP) ::  IniAngle, UR1, UR2, UR3, gamma_Bead, D_Bead,&
 XCM, YCM
 REAL(KIND = DP), DIMENSION(MaxAreaNum) :: AreaOriginX, AreaOriginY,&
 AreaOriginUx, AreaOriginUy
 REAL(KIND = DP), DIMENSION(NumBeads) :: XI, YI, ZI, XI_temp, &
 YI_temp, ZI_temp, FIx, FIy, FIz, NormRandVector4Beads, &
 XINormRandVector4Beads, YINormRandVector4Beads, &
 ZINormRandVector4Beads
 REAL(KIND = DP), DIMENSION(MaxNumMotors) :: XM, YM, ZM, Elongation
 REAL(KIND = DP), DIMENSION(NumFilament, MaxNumMotors) :: &
 ContactState, TempContact
 REAL(KIND = DP), DIMENSION(MaxNumMotors) :: F_Motor_X, F_Motor_Y, &
 F_Motor_Z
 REAL(KIND = DP), DIMENSION(NumBeads, NumFilament) :: X, Y, Z, Fx, &
 Fy, Fz, Force_Bead_X, Force_Bead_Y, Force_Bead_Z, &
 XNormRandVector4Beads, YNormRandVector4Beads, ZNormRandVector4Beads
 LOGICAL :: StateConstraint, StatConfinement
 LOGICAL, DIMENSION(MaxNumMotors) :: MotorStateUpdate
 CHARACTER(LEN=30) :: OutFileName
 seed_Assay = seed
 !---Calculating Parameters---
 gamma_Bead = 3.0_DP*pi*0.001_DP*BondLength/ &
 DLOG(BondLength/DiameterAF)
 D_Bead = kBT/gamma_Bead
 !---Output chamber boundary in vtk format---
 OPEN(10,FILE='InitialCondition.txt')
 !---Repeat Assay---
 Loop_Assay: DO I_Assay=1, NumAssay
 !---Assay Initial Condition---
 CALL sgrnd(seed_Assay)
 TS = 0
 ActiveMotorIdxOffset = 0
 ActiveMotorIdxEnd = 0
 AreaCounter = 1
 EraseCounter = 0
 OutputfileCounter = 0
 XM = -100.0_DP
 YM = 5.0_DP
 ZM = 0.0_DP
 Release_ADP = 0
 IniAngle = 0.0_DP*pi
 WRITE(10,'(I3,I7,F15.10)') I_Assay, seed_Assay, IniAngle
 seed_Assay = seed_Assay+1
 !---OPEN OUTPUT FILE---
 WRITE(OutFileName,'(A,I3.3,A)') 'ForCheck_A', I_Assay, '.txt'
 !---Output file for checking the program---
 OPEN(20,FILE=OutFileName)
 !---Output file for checking the program---
 UnitNumConformation = 11
 WRITE(OutFileName,'(A,I3.3,A)') 'Conformation_A', I_Assay, '.txt'
 OPEN(UnitNumConformation,FILE=OutFileName)
 UnitNumTipXY = 12
 WRITE(OutFileName,'(A,I3.3,A)') 'TipXY_A', I_Assay, '.txt'
 OPEN(UnitNumTipXY,FILE=OutFileName)
 UnitNumMotorStates = 13
 WRITE(OutFileName,'(A,I3.3,A)') 'MotorStates_A', I_Assay, '.txt'
```

```
OPEN(UnitNumMotorStates,FILE=OutFileName)
UnitNumAreaEraseCounter = 14
WRITE(OutFileName,'(A,I3.3,A)') 'AreaEraseCounter_A', I_Assay, '.txt'
OPEN(UnitNumAreaEraseCounter,FILE=OutFileName)
!---Initial Conformation of Filaments---
DO I=1, NumFilament
  X(NumBeads,I) = 0.0_DP
  Y(NumBeads,I) = 0.0_DP
  Z = 0.0125_DP
  DO J=NumBeads-1, 1, -1
    X(J,I) = X(J+1,I) + BondLength*DCOS(IniAngle)
    Y(J,I) = Y(J+1,I) + BondLength*DSIN(IniAngle)
  END DO
END DO
!---Initial Motor Location---
CALL InitialMotorLocations(ActiveMotorIdxOffset, ActiveMotorIdxEnd, &
IniAngle, X, Y, XM, YM, ZM, ContactState, AddedMotorNum, &
AreaCounter, AreaOriginX, AreaOriginY, AreaOriginUx, AreaOriginUy, &
MotorType)
!---Filament-Motor Contact---
TempContact = 0.0_DP
DO IM=ActiveMotorIdxOffset+1, ActiveMotorIdxEnd
  CALL CheckMotorFilamentProximity(X, Y, Z, XM(IM), YM(IM), ZM(IM), &
  ContactState(:,IM), TempContact(:,IM))
END DO
CALL InitialMotorBinding(ActiveMotorIdxOffset, ActiveMotorIdxEnd, &
ContactState, TempContact)
!---Calculating Forces upon Motors and Beads---
CALL CalculateForceMotor(ActiveMotorIdxOffset, ActiveMotorIdxEnd, &
X, Y, Z, XM, YM, ZM, ContactState, F_Motor_X, F_Motor_Y, &
F_Motor_Z, Elongation)
CALL CalculateForceBead(ActiveMotorIdxOffset, ActiveMotorIdxEnd, &
ContactState, Force_Bead_X, Force_Bead_Y, Force_Bead_Z, F_Motor_X, &
F_Motor_Y, F_Motor_Z)
!---Calculating Forces upon Beads---
Fx=Force_Bending(X) + Force_Bead_X + ExtF_X(TS, X, Y, Z)
Fy=Force_Bending(Y) + Force_Bead_Y + ExtF_Y(TS, X, Y, Z)
Fz=Force_Bending(Z) + Force_Bead_Z + ExtF_Z(TS, X, Y, Z)
!---Output Area & Erase Counters---
WRITE(UnitNumAreaEraseCounter,*) TS, AreaCounter, EraseCounter
!---Various Outputs---
CALL OutputConformation(UnitNumConformation,TS,X,Y,Z)
CALL OutputTipXY(UnitNumTipXY,TS,X,Y)
CALL OutputMotorStates(UnitNumMotorStates, TS, &
ActiveMotorIdxOffset, ActiveMotorIdxEnd, X, Y, Z, XM, YM, ZM, &
F_Motor_X, F_Motor_Y, F_Motor_Z, ContactState, MotorType)
CALL OutputFilamentVTK(I_Assay, OutputfileCounter, X, Y, Z)
CALL OutputFilamentPlusEndVTK(I_Assay, OutputfileCounter, X, Y, Z)
CALL OutputContactState(I_Assay, OutputfileCounter, &
ActiveMotorIdxOffset, ActiveMotorIdxEnd, MotorType, ContactState)
CALL OutputIntMotorsVTK(I_Assay, OutputfileCounter, &
ActiveMotorIdxOffset, ActiveMotorIdxEnd, MotorType, &
ContactState, XM, YM, ZM)
CALL OutputMotorsVTK(I_Assay, OutputfileCounter, &
ActiveMotorIdxOffset, ActiveMotorIdxEnd, MotorType, XM, YM, ZM)
!---End of Initial Setting---
!---Start of Time Evolution---
Loop_Time: DO TS=1, NumTimeStep
!---Update of Motor States---
TempContact = 0.0_DP
MotorStateUpdate = .FALSE.
DO IM=ActiveMotorIdxOffset+1, ActiveMotorIdxEnd
!---Forced Detachment of Bound Motors---
  IF (ContactState(1,IM) >= 1.0_DP) THEN
    CALL MotorForcedDetachment(F_Motor_X(IM), F_Motor_Y(IM), &
    F_Motor_Z(IM), ContactState(:,IM), Release_ADP(IM), &
    Elongation(IM), MotorType(IM), MotorStateUpdate(IM))
!---Filament-Motor Binding---
  ELSE IF (NINT(ContactState(1,IM)) == 0) THEN
    CALL CheckMotorFilamentProximity(X, Y, Z, XM(IM), YM(IM), &
    ZM(IM), ContactState(:,IM), TempContact(:,IM))
    CALL MotorBinding(ContactState(:,IM), TempContact(:,IM), &
    MotorStateUpdate(IM))
    IF (MotorType(IM) == 1) THEN
      CALL MotorStep(ContactState(:,IM), TempContact(:,IM))
    ELSE IF (MotorType(IM) == 2) THEN
      CALL MotorStuck(ContactState(:,IM), TempContact(:,IM))
    END IF
  END IF
END DO
!---State Conversion of Motors---
DO IM=ActiveMotorIdxOffset+1, ActiveMotorIdxEnd
  IF ((MotorType(IM) == 1) .AND. (MotorStateUpdate(IM) == .FALSE.)) &
  THEN
    CALL MotorStateConv(X, Y, Z, F_Motor_X(IM), F_Motor_Y(IM), &
    F_Motor_Z(IM), ContactState(:,IM), Release_ADP(IM))
  END IF
END DO
!---Renew Motor Population---
IF (MOD(TS,AreaRenewDiv)==0) THEN
  CALL RenewMotorPopulation(ActiveMotorIdxOffset, &
  ActiveMotorIdxEnd, X, Y, XM, YM, ZM, ContactState, AddedMotorNum, &
  AreaCounter, EraseCounter, AreaOriginX, AreaOriginY, &
  AreaOriginUx, AreaOriginUy, MotorType)
END IF
!---Update of Filament States---
!---Generating RND Num Matrix---
DO I=1,NumFilament
  CALL NormRNDVector(NumBeads, NormRandVector4Beads)
  DO J=1,NumBeads
    XNormRandVector4Beads(J,I) = NormRandVector4Beads(J)
  END DO
END DO
DO I=1,NumFilament
  CALL NormRNDVector(NumBeads, NormRandVector4Beads)
  DO J=1,NumBeads
    YNormRandVector4Beads(J,I) = NormRandVector4Beads(J)
  END DO
END DO
DO I=1,NumFilament
  CALL NormRNDVector(NumBeads, NormRandVector4Beads)
  DO J=1,NumBeads
    ZNormRandVector4Beads(J,I) = NormRandVector4Beads(J)
  END DO
END DO
Loop_Filament: DO I=1, NumFilament
  DO J=1, NumBeads
    XI(J)=X(J,I)
  END DO
  DO J=1, NumBeads
    YI(J)=Y(J,I)
  END DO
  DO J=1, NumBeads
    ZI(J)=Z(J,I)
  END DO
  DO J=1, NumBeads
    FIx(J)=Fx(J,I)
  END DO
  DO J=1, NumBeads
    FIy(J)=Fy(J,I)
  END DO
  DO J=1, NumBeads
    FIz(J)=Fz(J,I)
  END DO
  DO J=1, NumBeads
    XINormRandVector4Beads(J)=XNormRandVector4Beads(J,I)
  END DO
  DO J=1, NumBeads
    YINormRandVector4Beads(J)=YNormRandVector4Beads(J,I)
  END DO
  DO J=1, NumBeads
    ZINormRandVector4Beads(J)=ZNormRandVector4Beads(J,I)
  END DO
!---Unconstrained Movements of Beads---
  XI_temp = XI + FIx/gamma_Bead*dt + XINormRandVector4Beads * &
  DSQRT(2.0_DP * D_Bead * dt)
  YI_temp = YI + FIy/gamma_Bead*dt + YINormRandVector4Beads * &
  DSQRT(2.0_DP * D_Bead * dt)
  ZI_temp = ZI + FIz/gamma_Bead*dt + ZINormRandVector4Beads * &
  DSQRT(2.0_DP * D_Bead * dt)
!---Iteration until Constraint and Confinement are Statisfied---
  NumIteration = 0
  StateConstraint = .FALSE.
  StatConfinement = .FALSE.
  IterationConstraintConfinement: DO
    IF (StateConstraint .AND. StatConfinement) EXIT &
    IterationConstraintConfinement
    IF (NumIteration > MaxNumIteration) THEN
      WRITE(*,*) "Too␣many␣iterations!"
      EXIT IterationConstraintConfinement
    END IF
    NumIteration = NumIteration + 1
!---Filament confinement---
    StatConfinement = .TRUE.
    Loop_Bead_Confinement: DO J=1, NumBeads
      CALL Confinement(XI_temp(J), YI_temp(J), ZI_temp(J), &
      XI(J), YI(J), ZI(J), StatConfinement)
    END DO Loop_Bead_Confinement
!---Constraint---
    StateConstraint = .TRUE.
    Loop_Bond_Constraint: DO J=1, NumBeads-1
      CALL Constraint(XI_temp(J), YI_temp(J), ZI_temp(J), &
      XI_temp(J+1), YI_temp(J+1), ZI_temp(J+1), XI(J), YI(J), &
      ZI(J), XI(J+1), YI(J+1), ZI(J+1), StateConstraint)
    END DO Loop_Bond_Constraint
  END DO IterationConstraintConfinement
!---Update of Position Variables---
  DO J=1,NumBeads
    X(J,I) = XI_temp(J)
  END DO
  DO J=1,NumBeads
    Y(J,I) = YI_temp(J)
  END DO
  DO J=1,NumBeads
    Z(J,I) = ZI_temp(J)
  END DO
END DO Loop_Filament
!---Calculating Forces upon Motors and dividing the Force to Beads---
CALL CalculateForceMotor(ActiveMotorIdxOffset, ActiveMotorIdxEnd, &
X, Y, Z, XM, YM, ZM, ContactState, F_Motor_X, F_Motor_Y, F_Motor_Z, &
Elongation)
CALL CalculateForceBead(ActiveMotorIdxOffset, ActiveMotorIdxEnd, &
ContactState, Force_Bead_X, Force_Bead_Y, Force_Bead_Z, F_Motor_X, &
F_Motor_Y, F_Motor_Z)
!---Calculating Total Forces upon Beads---
Fx=Force_Bending(X) + Force_Bead_X + ExtF_X(TS, X, Y, Z)
Fy=Force_Bending(Y) + Force_Bead_Y + ExtF_Y(TS, X, Y, Z)
Fz=Force_Bending(Z) + Force_Bead_Z + ExtF_Z(TS, X, Y, Z)
IF (MOD(TS,OutPutDiv2)==0) THEN
  CALL OutputMotorStates(UnitNumMotorStates, TS, &
  ActiveMotorIdxOffset, ActiveMotorIdxEnd, X, Y, Z, XM, YM, ZM, &
  F_Motor_X, F_Motor_Y, F_Motor_Z, ContactState, MotorType)
END IF
IF (MOD(TS,OutPutDiv)==0) THEN
  OutputfileCounter = OutputfileCounter + 1
!---Output Area & Erase Counters---
  WRITE(UnitNumAreaEraseCounter,*) TS, AreaCounter, EraseCounter
  !---Various Outputs Filament Conformation---
  CALL OutputConformation(UnitNumConformation,TS,X,Y,Z)
  CALL OutputTipXY(UnitNumTipXY,TS,X,Y)
  CALL OutputFilamentVTK(I_Assay, OutputfileCounter, X, Y, Z)
  CALL OutputFilamentPlusEndVTK(I_Assay, OutputfileCounter, X, Y, Z)
  CALL OutputContactState(I_Assay, OutputfileCounter, &
  ActiveMotorIdxOffset, ActiveMotorIdxEnd, MotorType, ContactState)
  CALL OutputIntMotorsVTK(I_Assay, OutputfileCounter, &
  ActiveMotorIdxOffset, ActiveMotorIdxEnd, MotorType, &
  ContactState, XM, YM, ZM)
  CALL OutputMotorsVTK(I_Assay, OutputfileCounter, &
  ActiveMotorIdxOffset, ActiveMotorIdxEnd, MotorType, XM, YM, ZM)
END IF
```

```
  IF (X(1,1) > XLimit) EXIT Loop_Time
  END DO Loop_Time
  CLOSE(UnitNumConformation)
  CLOSE(UnitNumTipXY)
  CLOSE(UnitNumMotorStates)
  CLOSE(UnitNumAreaEraseCounter)
  END DO Loop_Assay
  CLOSE(10)
END PROGRAM MAIN
```

A.3 Generate Normal Random Numbers

```
 SUBROUTINE NormRNDNum(NRN)
  USE PARAMETERS, ONLY : DP
  USE mtmod, ONLY : grnd !From a separate program
  REAL(KIND = DP), INTENT(OUT) :: NRN
  REAL(KIND = DP) :: UR1, UR2
  UR1 = grnd()
  IF (UR1 == 0.0_DP) UR1 = grnd()
  UR2 = grnd()
  NRN = DSQRT(-2.0_DP*DLOG(UR1)) * DCOS(2.0_DP*pi*(UR2))
END SUBROUTINE NormRNDNum
```

A.4 Bending Force Calculation

```
 FUNCTION Force_Bending(Coordinate)
  USE PARAMETERS, ONLY : DP, NumFilament, NumBeads, BondLength, EI
  REAL(KIND = DP), DIMENSION(NumBeads, NumFilament) :: Force_Bending
  REAL(KIND = DP), DIMENSION(NumBeads, NumFilament), &
  INTENT(IN) :: Coordinate
  INTEGER :: I, J
  REAL(KIND = DP) :: F
  Force_Bending = 0.0_DP
  DO I=1, NumFilament
    DO J=2, NumBeads-1
      F = Coordinate(J+1,I) - 2.0_DP*Coordinate(J,I) + &
      Coordinate(J-1,I)
      Force_Bending(J-1,I) = Force_Bending(J-1,I) + (-1.0_DP)*F
      Force_Bending(J,I) = Force_Bending(J,I) + 2.0_DP*F
      Force_Bending(J+1,I) = Force_Bending(J+1,I) + (-1.0_DP)*F
    END DO
  END DO
  Force_Bending = Force_Bending*EI/(BondLength**3)
END FUNCTION Force_Bending
```

A.5 Initial Myosin Binding

```
 SUBROUTINE InitialMotorBinding(ActiveMotorIdxOffset, &
ActiveMotorIdxEnd, ContactState, TempContact)
  USE PARAMETERS , ONLY : DP, NumFilament, NumBeads
  USE mtmod, ONLY : grnd
  INTEGER(KIND = Range15), INTENT(IN):: ActiveMotorIdxOffset, &
  ActiveMotorIdxEnd
  REAL(KIND = DP), DIMENSION(NumFilament, MaxNumMotors), &
  INTENT(INOUT) :: ContactState, TempContact
  INTEGER(KIND = Range15) :: IM
  INTEGER :: II
  REAL(KIND = DP) ::  UR
  DO IM=ActiveMotorIdxOffset+1, ActiveMotorIdxEnd
    DO II=1, NumFilament
      IF ((TempContact(II, IM) >= 1.0_DP) .AND. &
      (TempContact(II, IM) <= DBLE(NumBeads))) THEN
      UR = grnd()
        IF(UR <= 0.1_DP) THEN
          !---Closer to steady state population of binding motors---
          ContactState(II, IM) = TempContact(II, IM)
          TempContact(II, IM) = 0.0_DP
        END IF
      END IF
    END DO
  END DO
END SUBROUTINE InitialMotorBinding
```

A.6 Check Myosin-Actin Proximity

```
 SUBROUTINE CheckMotorFilamentProximity(X, Y, Z, XM_IM, YM_IM, &
ZM_IM, ContactState_IM, TempContact_IM)
  USE PARAMETERS, ONLY : DP, MaxNumMotors, NumFilament, NumBeads, &
  BondLength, CaptureRadius
  REAL(KIND = DP), DIMENSION(NumBeads, NumFilament), &
  INTENT(IN) :: X, Y, Z
  REAL(KIND = DP), INTENT(IN) :: XM_IM, YM_IM, ZM_IM
  REAL(KIND = DP), DIMENSION(NumFilament), &
  INTENT(IN) :: ContactState_IM
  REAL(KIND = DP), DIMENSION(NumFilament), &
  INTENT(OUT) :: TempContact_IM
  REAL(KIND = DP) :: Intercept, SqDistance
  INTEGER(KIND = Range15) :: IM
  INTEGER :: II, JJ
  Loop_Filament: DO II=1, NumFilament
    DO JJ=1, NumBeads-1
      Intercept = (XM_IM - X(JJ,1))*(X(JJ+1,1) - X(JJ,1)) + &
            (YM_IM - Y(JJ,1))*(Y(JJ+1,1) - Y(JJ,1)) + &
            (ZM_IM - Z(JJ,1))*(Z(JJ+1,1) - Z(JJ,1))
      Intercept = Intercept/BondLength
      SqDistance = (XM_IM - X(JJ,1))*(XM_IM - X(JJ,1)) + &
            (YM_IM - Y(JJ,1))*(YM_IM - Y(JJ,1)) + &
            (ZM_IM - Z(JJ,1))*(ZM_IM - Z(JJ,1)) - Intercept**2
      IF ((SqDistance <= CaptureRadius**2) .AND.
      (Intercept >= 0.0_DP) .AND. (Intercept <= BondLength)) THEN
        TempContact_IM(II) = DBLE(JJ) + Intercept/BondLength
      END IF
    END DO
  END DO Loop_Filament
END SUBROUTINE CheckMotorFilamentProximity
```

A.7 Binding Myosin

```
 SUBROUTINE MotorBinding(ContactState_IM, TempContact_IM, &
MotorStateUpdate_IM)
  USE PARAMETERS , ONLY : DP, NumFilament, dt, k_a
  USE mtmod, ONLY : grnd
  REAL(KIND = DP), DIMENSION(NumFilament), &
  INTENT(INOUT) :: ContactState_IM, TempContact_IM
  LOGICAL, INTENT(OUT) :: MotorStateUpdate_IM
  REAL(KIND = DP) :: UR
  INTEGER :: II
  UR = grnd()
  Loop_Filament_ContactCheck: DO II=1, NumFilament
    IF ((TempContact_IM(II) >= 1.0_DP ) .AND. &
    (NINT(ContactState_IM(II)) == 0)) THEN
      IF(UR > k_a*dt)THEN
        TempContact_IM(II) = 0.0_DP
      ELSE
        MotorStateUpdate_IM = .TRUE.
      END IF
    END IF
  END DO Loop_Filament_ContactCheck
END SUBROUTINE MotorBinding
```

A.8 Myosin Step

```
 SUBROUTINE MotorStep(ContactState_IM, TempContact_IM)
  USE PARAMETERS , ONLY : DP, NumFilament, dt, k_a
  USE mtmod, ONLY : grnd
  REAL(KIND = DP), DIMENSION(NumFilament), &
  INTENT(INOUT) :: ContactState_IM, TempContact_IM
  INTEGER :: II
  Loop_Filament_ContactCheck: DO II=1, NumFilament
    IF (TempContact_IM(II) >= 1.0_DP )THEN
      ContactState_IM(II) = TempContact_IM(II) + Stepsize/BondLength
    END IF
  END DO Loop_Filament_ContactCheck
END SUBROUTINE MotorStep
```

A.9 Myosin Getting Stuck

```
 SUBROUTINE MotorStuck(ContactState_IM, TempContact_IM)
  USE PARAMETERS , ONLY : DP, NumFilament, dt, k_a
  USE mtmod, ONLY : grnd
  REAL(KIND = DP), DIMENSION(NumFilament), INTENT(INOUT) :: &
  ContactState_IM, TempContact_IM
  INTEGER :: II
  Loop_Filament_ContactCheck: DO II=1, NumFilament
    IF (TempContact_IM(II) >= 1.0_DP )THEN
      ContactState_IM(II) = TempContact_IM(II)
    END IF
  END DO Loop_Filament_ContactCheck
END SUBROUTINE MotorStuck
```

A.10 Force Actin on Myosin

```
 SUBROUTINE CalculateForceMotor(ActiveMotorIdxOffset, &
ActiveMotorIdxEnd, X, Y, Z, XM, YM, ZM, ContactState, &
F_Motor_X, F_Motor_Y, F_Motor_Z, Elongation)
  USE PARAMETERS, ONLY : DP, MaxNumMotors, NumFilament, &
  CaptureRadius, k
  REAL(KIND = DP), DIMENSION(NumBeads, NumFilament), &
  INTENT(IN) :: X, Y, Z
  REAL(KIND = DP), DIMENSION(MaxNumMotors), &
  INTENT(IN) :: XM, YM, ZM
  REAL(KIND = DP), DIMENSION(NumFilament, MaxNumMotors), &
  INTENT(IN) :: ContactState
  REAL(KIND = DP), DIMENSION(MaxNumMotors), &
  INTENT(OUT) :: F_Motor_X, F_Motor_Y, F_Motor_Z
  INTEGER(KIND = Range15), INTENT(IN):: ActiveMotorIdxOffset, &
  ActiveMotorIdxEnd
  REAL(KIND = DP) :: Intercept_Contact, X_Contact, Y_Contact, &
  Z_Contact
  REAL(KIND = DP), DIMENSION(MaxNumMotors), &
  INTENT(OUT) :: Elongation
  INTEGER(KIND = Range15) :: IM
  INTEGER :: II, J_Contact
  F_Motor_X = 0.0_DP
  F_Motor_Y = 0.0_DP
  F_Motor_Z = 0.0_DP
  DO IM=ActiveMotorIdxOffset+1, ActiveMotorIdxEnd
    IF (ContactState(1,IM) >= 1.0_DP) THEN
      J_Contact = INT(ContactState(1,IM))
      Intercept_Contact = ContactState(1,IM) - DBLE(J_Contact)
      IF (J_Contact >= NumBeads) THEN
        X_Contact = X(NumBeads,1) + Intercept_Contact * &
        (X(NumBeads,1) - X(NumBeads-1,1))
        Y_Contact = Y(NumBeads,1) + Intercept_Contact * &
        (Y(NumBeads,1) - Y(NumBeads-1,1))
        Z_Contact = Z(NumBeads,1) + Intercept_Contact * &
        (Z(NumBeads,1) - Z(NumBeads-1,1))
      ELSE
```

```
        X_Contact = X(J_Contact,1) + Intercept_Contact * &
        (X(J_Contact+1,1) - X(J_Contact,1))
        Y_Contact = Y(J_Contact,1) + Intercept_Contact * &
        (Y(J_Contact+1,1) - Y(J_Contact,1))
        Z_Contact = Z(J_Contact,1) + Intercept_Contact * &
        (Z(J_Contact+1,1) - Z(J_Contact,1))
      END IF
      Elongation(IM) = (XM(IM) - X_Contact)**2 + &
      (YM(IM) - Y_Contact)**2 + (ZM(IM) - Z_Contact)**2
      Elongation(IM) = DSQRT(Elongation(IM))
      F_Motor_X(IM) = k * Elongation(IM) * &
      (XM(IM) - X_Contact)/Elongation(IM)
      F_Motor_Y(IM) = k * Elongation(IM) * &
      (YM(IM) - Y_Contact)/Elongation(IM)
      F_Motor_Z(IM) = k * Elongation(IM) * &
      (ZM(IM) - Z_Contact)/Elongation(IM)
    END IF
  END DO
END SUBROUTINE CalculateForceMotor
```

A.11 Forced Myosin Detachment

```
 SUBROUTINE MotorForcedDetachment(F_Motor_X_IM, F_Motor_Y_IM, &
F_Motor_Z_IM, ContactState_IM, Release_ADP_IM, Elongation_IM, &
MotorType_IM, MotorStateUpdate_IM)
  USE PARAMETERS, ONLY : DP, MaxNumMotors, NumFilament, k, &
  F_Motor_Detach1, F_Motor_Detach2
  INTEGER, INTENT(IN) :: MotorType_IM
  INTEGER, INTENT(INOUT) :: Release_ADP_IM
  REAL(KIND = DP), INTENT(INOUT) :: F_Motor_X_IM, F_Motor_Y_IM, &
  F_Motor_Z_IM
  REAL(KIND = DP), DIMENSION(NumFilament), &
  INTENT(INOUT) :: ContactState_IM
  REAL(KIND = DP), INTENT(IN) :: Elongation_IM
  LOGICAL, INTENT(OUT) :: MotorStateUpdate_IM
  INTEGER :: II
  SELECT CASE(MotorType_IM)
  CASE (1)
    IF (k*Elongation_IM >= F_Motor_Detach1) THEN
      F_Motor_X_IM = 0.0_DP
      F_Motor_Y_IM = 0.0_DP
      F_Motor_Z_IM = 0.0_DP
      Scan_Contact_Filament1: DO II=1, NumFilament
        IF (ContactState_IM(II) >= 1.0_DP) THEN
          ContactState_IM(II) = -1.0_DP
          Release_ADP_IM = 0
          MotorStateUpdate_IM = .TRUE.
        END IF
      END DO Scan_Contact_Filament1
    END IF
  CASE (2)
    IF (k*Elongation_IM >= F_Motor_Detach2) THEN
      F_Motor_X_IM = 0.0_DP
      F_Motor_Y_IM = 0.0_DP
      F_Motor_Z_IM = 0.0_DP
      Scan_Contact_Filament2: DO II=1, NumFilament
        IF (ContactState_IM(II) >= 1.0_DP) THEN
          ContactState_IM(II) = 0.0_DP
        END IF
      END DO Scan_Contact_Filament2
    END IF
  END SELECT
END SUBROUTINE MotorForcedDetachment
```

A.12 Force Acting on Actin Beads

```
 SUBROUTINE CalculateForceBead(ActiveMotorIdxOffset, &
ActiveMotorIdxEnd, ContactState, Force_Bead_X, Force_Bead_Y, &
Force_Bead_Z, F_Motor_X, F_Motor_Y, F_Motor_Z)
  USE PARAMETERS, ONLY : DP, MaxNumMotors, NumFilament
  REAL(KIND = DP), DIMENSION(NumFilament, MaxNumMotors), &
  INTENT(IN) :: ContactState
  REAL(KIND = DP), DIMENSION(NumBeads, NumFilament), &
  INTENT(OUT) :: Force_Bead_X, Force_Bead_Y, Force_Bead_Z
  REAL(KIND = DP), DIMENSION(MaxNumMotors), &
  INTENT(IN) :: F_Motor_X, F_Motor_Y, F_Motor_Z
  INTEGER(KIND = Range15), INTENT(IN):: ActiveMotorIdxOffset, &
  ActiveMotorIdxEnd
  REAL(KIND = DP) :: Intercept_Contact
  INTEGER(KIND = Range15) :: IM
  INTEGER :: II, J_Contact
  Force_Bead_X = 0.0_DP
  Force_Bead_Y = 0.0_DP
  Force_Bead_Z = 0.0_DP
  DO IM=ActiveMotorIdxOffset+1, ActiveMotorIdxEnd
    Scan_Contact_Filament: DO II=1, NumFilament
      IF (ContactState(II,IM) >= 1.0_DP) THEN
        J_Contact = INT(ContactState(II,IM))
        Intercept_Contact = ContactState(II,IM) - DBLE(J_Contact)
        IF (J_Contact >= NumBeads) THEN
          Force_Bead_X(NumBeads,II) = Force_Bead_X(NumBeads,II) + &
          F_Motor_X(IM)
          Force_Bead_Y(NumBeads,II) = Force_Bead_Y(NumBeads,II) + &
          F_Motor_Y(IM)
          Force_Bead_Z(NumBeads,II) = Force_Bead_Z(NumBeads,II) + &
          F_Motor_Z(IM)
        ELSE
          Force_Bead_X(J_Contact,II) = Force_Bead_X(J_Contact,II) + &
          (1.0_DP - Intercept_Contact)*F_Motor_X(IM)
          Force_Bead_Y(J_Contact,II) = Force_Bead_Y(J_Contact,II) + &
          (1.0_DP - Intercept_Contact)*F_Motor_Y(IM)
          Force_Bead_Z(J_Contact,II) = Force_Bead_Z(J_Contact,II) + &
          (1.0_DP - Intercept_Contact)*F_Motor_Z(IM)
          Force_Bead_X(J_Contact+1,II) = Force_Bead_X(J_Contact+1, &
          II) + Intercept_Contact*F_Motor_X(IM)
          Force_Bead_Y(J_Contact+1,II) = Force_Bead_Y(J_Contact+1, &
          II) + Intercept_Contact*F_Motor_Y(IM)
          Force_Bead_Z(J_Contact+1,II) = Force_Bead_Z(J_Contact+1, &
          II) + Intercept_Contact*F_Motor_Z(IM)
        END IF
      END IF
    END DO Scan_Contact_Filament
  END DO
END SUBROUTINE CalculateForceBead
```

A.13 Constraints

```
 SUBROUTINE Constraint(XItempJ0, YItempJ0, ZItempJ0, XItempJ1, &
YItempJ1, ZItempJ1, XIJ0, YIJ0, ZIJ0, XIJ1, YIJ1, ZIJ1, &
StateConstraint)
  USE PARAMETERS, ONLY : DP, Tol, BondLength
  REAL(KIND = DP), INTENT(INOUT) :: XItempJ0, XItempJ1, YItempJ0, &
  YItempJ1, ZItempJ0, ZItempJ1
  REAL(KIND = DP), INTENT(IN) :: XIJ0, XIJ1, YIJ0, YIJ1, ZIJ0, ZIJ1
  LOGICAL, INTENT(INOUT) :: StateConstraint
  REAL(KIND = DP) :: XI_tempAB, YI_tempAB, ZI_tempAB, XIAB, YIAB, &
  ZIAB, BeadsDisSq, DiffSq, gAB, DX, DY, DZ
  XI_tempAB = XItempJ1 - XItempJ0
  YI_tempAB = YItempJ1 - YItempJ0
  ZI_tempAB = ZItempJ1 - ZItempJ0
  XIAB = XIJ1 - XIJ0
  YIAB = YIJ1 - YIJ0
  ZIAB = ZIJ1 - ZIJ0
  BeadsDisSq = XI_tempAB**2 + YI_tempAB**2 + ZI_tempAB**2
  DiffSq = BondLength**2 - BeadsDisSq
  IF (DABS(DiffSq) > 2.0_DP*Tol*BondLength**2) THEN
    StateConstraint = .FALSE.
    gAB = DiffSq / 4.0_DP / (XI_tempAB*XIAB + YI_tempAB*YIAB + &
    ZI_tempAB*ZIAB)
    DX = gAB*XIAB
    DY = gAB*YIAB
    DZ = gAB*ZIAB
    XItempJ0 = XItempJ0 - DX
    YItempJ0 = YItempJ0 - DY
    ZItempJ0 = ZItempJ0 - DZ
    XItempJ1 = XItempJ1 + DX
    YItempJ1 = YItempJ1 + DY
    ZItempJ1 = ZItempJ1 + DZ
  END IF
END SUBROUTINE Constraint
```

A.14 Renew the Myosin Population

```
 SUBROUTINE RenewMotorPopulation(ActiveMotorIdxOffset, &
ActiveMotorIdxEnd, X, Y, XM, YM, ZM, ContactState, AddedMotorNum, &
AreaCounter, EraseCounter, AreaOriginX, AreaOriginY, AreaOriginUx, &
AreaOriginUy, MotorType)
  USE PARAMETERS, ONLY : Range15, DP, MaxNumMotors, NumFilament, &
  NumBeads, BondLength, HorizontalLength, VerticalLength, &
  Motor_Density, Type1Ratio
  INTEGER, DIMENSION(MaxNumMotors), INTENT(INOUT) :: MotorType
  REAL(KIND = DP), DIMENSION(NumBeads, NumFilament), &
  INTENT(IN) :: X, Y
  REAL(KIND = DP), DIMENSION(MaxNumMotors), INTENT(INOUT) :: XM, YM, ZM
  REAL(KIND = DP), DIMENSION(NumFilament, MaxNumMotors), &
  INTENT(INOUT) :: ContactState
  INTEGER(KIND = Range15), DIMENSION(MaxAreaNum), &
  INTENT(INOUT) :: AddedMotorNum
  INTEGER, INTENT(INOUT)  :: AreaCounter, EraseCounter
  REAL(KIND = DP), DIMENSION(MaxAreaNum), INTENT(INOUT) :: &
  AreaOriginX, AreaOriginY, AreaOriginUx, AreaOriginUy
  INTEGER(KIND = Range15), INTENT(INOUT) :: ActiveMotorIdxOffset, &
  ActiveMotorIdxEnd
  INTEGER(KIND = Range15):: IM, AddedMotorCounter
  INTEGER :: I_Area, JJ, JJJ
  REAL(KIND = DP) :: XCM, YCM, XM_New, YM_New, UR1, UR2, UR3
  LOGICAL :: NearBoundary, InNewArea, OutOldArea
  !---Erase Old Motor Region---
  OutOldArea = .TRUE.
  DO JJJ=1, NumBeads
    IF ((DABS((X(JJJ,1)-AreaOriginX(EraseCounter+1))*AreaOriginUx(&
    EraseCounter+1) + (Y(JJJ,1)-AreaOriginY(EraseCounter+1))* &
    AreaOriginUy(EraseCounter+1)) <= 0.5_DP*HorizontalLength + &
    0.5_DP) .AND. (DABS(-(X(JJJ,1)-AreaOriginX(EraseCounter+1))* &
    AreaOriginUy(EraseCounter+1) + (Y(JJJ,1)-AreaOriginY(EraseCounter &
    +1))* AreaOriginUx(EraseCounter+1)) <= 0.5_DP*VerticalLength + &
    0.5_DP)) THEN
      OutOldArea = .FALSE.
    END IF
  END DO
  IF (OutOldArea) THEN
    EraseCounter = EraseCounter + 1
    ActiveMotorIdxOffset = ActiveMotorIdxOffset + AddedMotorNum(&
    EraseCounter)
  END IF
  !---check if beads are near boundary---
  XCM=SUM(X(:,1))/DBLE(NumBeads)
  YCM=SUM(Y(:,1))/DBLE(NumBeads)
  NearBoundary = .FALSE.
  DO JJ=1, NumBeads
    IF (DABS((X(JJ,1)-AreaOriginX(AreaCounter))*AreaOriginUx(&
    AreaCounter) + (Y(JJ,1)-AreaOriginY(AreaCounter))*AreaOriginUy(&
    AreaCounter)) >= 0.5_DP*HorizontalLength - 0.5_DP*DBLE(&
    NumBeads-1)* BondLength - 1.0_DP) NearBoundary = .TRUE.
    IF (DABS(-(X(JJ,1)-AreaOriginX(AreaCounter))*AreaOriginUy(&
    AreaCounter) + (Y(JJ,1)-AreaOriginY(AreaCounter))*AreaOriginUx(&
    AreaCounter)) >= 0.5_DP* VerticalLength - 0.5_DP*DBLE(&
    NumBeads-1)* BondLength - 1.0_DP) NearBoundary = .TRUE.
  END DO
  IF (NearBoundary == .TRUE.) THEN
  !---Generate New Motor Region---
    AreaCounter = AreaCounter + 1
    AreaOriginX(AreaCounter) = XCM
    AreaOriginY(AreaCounter) = YCM
    AreaOriginUx(AreaCounter) = (X(1,1) - X(2,1))/DSQRT((X(1,1) - &
    X(2,1))* (X(1,1) - X(2,1)) + (Y(1,1) - Y(2,1))*(Y(1,1) - Y(2,1)))
    AreaOriginUy(AreaCounter) = (Y(1,1) - Y(2,1))/DSQRT((X(1,1) - &
    X(2,1))* (X(1,1) - X(2,1)) + (Y(1,1) - Y(2,1))*(Y(1,1) - Y(2,1)))
    AddedMotorCounter = 0
    DO IM=1, NINT(Motor_Density*HorizontalLength*VerticalLength, &
    Range15)
      UR1 = grnd()
      UR2 = grnd()
      XM_New = XCM + HorizontalLength * (UR1 - 0.5_DP) * AreaOriginUx(&
      AreaCounter) - VerticalLength * (UR2 - 0.5_DP) * AreaOriginUy(&
      AreaCounter)
      YM_New = YCM + HorizontalLength * (UR1 - 0.5_DP) * AreaOriginUy(&
      AreaCounter) + VerticalLength * (UR2 - 0.5_DP) * AreaOriginUx(&
      AreaCounter)
      InNewArea = .TRUE.
      IF (AreaCounter == 1) THEN
        IF ((DABS((XM_New-AreaOriginX(AreaCounter))*AreaOriginUx(&
        AreaCounter) + (YM_New-AreaOriginY(AreaCounter))*AreaOriginUy(&
        AreaCounter)) <= 0.5_DP*HorizontalLength) .AND. &
        (DABS(-(XM_New-AreaOriginX(AreaCounter))*AreaOriginUy(&
        AreaCounter) + (YM_New-AreaOriginY(AreaCounter))*AreaOriginUx(&
        AreaCounter)) <= 0.5_DP*VerticalLength)) THEN
          InNewArea = .FALSE.
        END IF
      ELSE
        DO I_Area = EraseCounter+1, AreaCounter-1
          IF ((DABS((XM_New-AreaOriginX(I_Area))*AreaOriginUx(I_Area) &
          + (YM_New-AreaOriginY(I_Area))*AreaOriginUy(I_Area)) <= &
          0.5_DP*HorizontalLength).AND.(DABS(-(XM_New-AreaOriginX(&
          I_Area))* AreaOriginUy(I_Area) + (YM_New-AreaOriginY(&
          I_Area))* AreaOriginUx(I_Area)) <= 0.5_DP*VerticalLength)) THEN
            InNewArea = .FALSE.
          END IF
        END DO
      END IF
      IF (InNewArea == .TRUE.) THEN
        AddedMotorCounter = AddedMotorCounter + 1
        XM(ActiveMotorIdxEnd + AddedMotorCounter) = XM_New
        YM(ActiveMotorIdxEnd + AddedMotorCounter) = YM_New
        ZM(ActiveMotorIdxEnd + AddedMotorCounter) = 0.0_DP
        UR1 = grnd()
        IF (UR1 <= Type1Ratio) THEN
          MotorType(ActiveMotorIdxEnd + AddedMotorCounter) = 1
          UR2 = grnd()
          IF(UR2 <= 0.091) THEN
            ContactState(:,ActiveMotorIdxEnd + AddedMotorCounter) = &
            -1.0_DP
          ELSE
            ContactState(:,ActiveMotorIdxEnd + AddedMotorCounter) = &
            0.0_DP
          END IF
        ELSE
          MotorType(ActiveMotorIdxEnd + AddedMotorCounter) = 2
          ContactState(:,ActiveMotorIdxEnd + AddedMotorCounter) = 0.0_DP
        END IF
      END IF
    END DO
    AddedMotorNum(AreaCounter) = AddedMotorCounter
    ActiveMotorIdxEnd = ActiveMotorIdxEnd + AddedMotorCounter
  END IF
```

```
END SUBROUTINE RenewMotorPopulation
```

A.15 Myosin State Conversion

```
 SUBROUTINE MotorStateConv(X, Y, Z, F_Motor_X_IM, F_Motor_Y_IM, &
F_Motor_Z_IM, ContactState_IM, Release_ADP_IM)
  USE PARAMETERS
  USE mtmod, ONLY : grnd
  REAL(KIND = DP), DIMENSION(NumBeads, NumFilament), &
  INTENT(IN) :: X, Y, Z
  INTEGER, INTENT(INOUT) :: Release_ADP_IM
  REAL(KIND = DP), DIMENSION(NumFilament), &
  INTENT(INOUT) :: ContactState_IM
  REAL(KIND = DP), DIMENSION(NumFilament), INTENT(IN) :: &
  F_Motor_X_IM, F_Motor_Y_IM, F_Motor_Z_IM
  REAL(KIND = DP) :: UR, F_Motor_Tangent, k_d
  INTEGER(KIND = Range15) :: IM
  INTEGER :: II, J_Contact
  DO II=1, NumFilament
    J_Contact = INT(ContactState_IM(II))
    UR = grnd()
    SELECT CASE(J_Contact)
    CASE (-1)
      IF(UR <= k_hp*dt)THEN
        ContactState_IM(II) = 0.0_DP
      END IF
    CASE (0)
      IF(UR <= k_hm*dt)THEN
        ContactState_IM(II) = -1.0_DP
      END IF
    CASE (1:)
      IF (Release_ADP_IM == 0) THEN
        IF (J_Contact >= NumBeads) THEN
          F_Motor_Tangent = (F_Motor_X_IM(II)*(X(J_Contact,II) - &
          X(J_Contact-1,II))+ F_Motor_Y_IM(II)*(Y(J_Contact,II) - &
          Y(J_Contact-1,II))+ F_Motor_Z_IM(II)*(Z(J_Contact,II) - &
          Z(J_Contact-1,II)))/BondLength
        ELSE
          F_Motor_Tangent = (F_Motor_X_IM(II)*(X(J_Contact+1,II) - &
          X(J_Contact,II))+ F_Motor_Y_IM(II)*(Y(J_Contact+1,II) - &
          Y(J_Contact,II))+ F_Motor_Z_IM(II)*(Z(J_Contact+1,II) - &
          Z(J_Contact,II)))/BondLength
        END IF
        k_d = k_d0*DEXP(-F_Motor_Tangent*delta_x/KBT)
        IF(UR <= k_d*dt)THEN
          Release_ADP_IM = 1
        END IF
      ELSE
        IF(UR <= k_t*ATP*dt)THEN
          Release_ADP_IM = 0
          ContactState_IM(II) = -1.0_DP
        END IF
      END IF
    END SELECT
  END DO
END SUBROUTINE MotorStateConv
```

B Sample Analysis Programs

This section shows the sample analysis programs referenced in this dissertation.

B.1 Path Persistence Length Calculation

```
#!/usr/bin/env python
# coding: utf-8
# Input: trajectory x,y data
# Output: persistence length plots
#====================================================================

# Import important libs
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import os, glob
from scipy.optimize import curve_fit
plt.style.use("ggplot")
cm = 1/2.54
#====================================================================

#Import trajectory data
#r = 1.00; v = 7.28 # Set: Active ratio and Filament speed
#r = 0.98; v = 6.10
#r = 0.96; v = 4.66
r = 0.94; v = 3.41
R = "{0:.2f}".format(r)
conff0 = glob.glob('data/Lp00042/ConfT5S**'+R+'.txt')
nm = 'R'+R+'-kBT0.0042'
conff0 = sorted(conff0); no = len(conff0); no = no-1
print("Imported␣data␣list:"); print(conff0)
#====================================================================

# Required parameters
beads = 13 # Filament beads (actin)
#jmp = 10 ; dt = 'dt0'; Dt = 0.1 # Adjust delta t
jmp = 20 ; dt = 'dt2'; Dt = 0.2 # For actin
#jmp = 30 ; dt = 'dt0'; Dt = 3 # For microtubule*
#====================================================================

# Calculate the difference and store the values
conf0 = [];
xy0 = []; xy_1 = []; xy1 = [];
xdiff0 = []; ydiff0 = [];
for i in conff0:
  _ = pd.read_csv(i, names=['t','x','y','z'], delim_whitespace=True)
  xy0_ = _[0::beads] # jump beads
  xy0.append(xy0_) # _
  _ = xy0_[0::jmp] # _
  _ = _.reset_index(drop=True)
  conf0.append(_)
  xdiff0.append(np.diff(_['x']))
  ydiff0.append(np.diff(_['y']))
#====================================================================

# Plot sample trajectories
fig, ax = plt.subplots(1,1, figsize=(10*cm, 10*cm), sharex = True, \
sharey = True)

for i in range(no+1):
  ax.plot(xy0[i]['x'], xy0[i]['y'], lw=1)
ax.minorticks_on()
ax.tick_params('both', direction='in', length=8, top=True, \
right=True, which='major')
ax.tick_params('both', direction='in', length=4, top=True, \
right=True, which='minor')
ax.set_xlabel('X($\mu␣m$)')
ax.set_ylabel('Y($\mu␣m$)')
ax.set_title('$r_{substrate}␣=␣%.2f$␣|␣$\Delta␣t␣=␣%.2f\
sec$'%(r,Dt/jmp), fontsize=14)
```

```
print("Trajectories:␣")
plt.savefig('fig/Traj-'+str(round(r,2))+'-Dt'+str(round(Dt,2))\
+nm+'.pdf', format='pdf', bbox_inches='tight')
plt.show()
#====================================================================


# Calculate unit vector
ubx0 = []; uby0 = []
ubx_1 = []; uby_1 = []
ubx1 = []; uby1 = []
for i in range(len(xdiff0)):
  b0 = np.sqrt(xdiff0[i]**2 + ydiff0[i]**2)
  ubx0.append(xdiff0[i]/b0)
  uby0.append(ydiff0[i]/b0)
#====================================================================

# Change numpy array to pandas
ub0 = []; ub_1 = []; ub1 = []
for i in range(len(ubx0)):
  ub0.append(pd.DataFrame({'ubx':ubx0[i], 'uby':uby0[i]}))
#====================================================================


# Calculate the dot product/correlations
_ = []; ds0 = []; dsm0 = []; s0 = []; c = 0;
ds0_ = []; dsm_ = []; s_ = []

# try:
#   os.remove('ds_ub_status.txt')
# except:
#   print("'ds_ub_status.txt' does not exist.")

for h in range(len(ub0)):
  for i in range(len(ub0[h])): # Save status for checking
#     print("Ds = %s"%i, file=open('ds_ub_status.txt','a'))
    for j in range(len(ub0[h])):
      try:
        _.append(np.dot(\
        ub0[h].loc[j].values,ub0[h].loc[j+i].values))
#         print("Ub%s.Ub%s"%(j,j+i), file=open('ds_ub_status.txt','a'))
      except:
        pass #print("No: Ub%s.Ub%s"%(j,j+i))
    ds0_.append(_) # not necessary to save or is it?
    dsm_.append(np.mean(_))
    s_.append(c)
    _ = [] # empty this bucket
    c+=1;
  s0.append(s_)
  dsm0.append(dsm_)
  ds0.append(ds0_)
  c = 0; ds0_ = []; dsm_ = []; s_ = [] # reset stuff

s0 = np.array(s0)
#ds0 = np.array(ds0)
dsm0 = np.array(dsm0)
#====================================================================


# Calculate the mean of the correlations
dfs0 = pd.DataFrame(s0.T)
dfdsm0 = pd.DataFrame(dsm0.T)
s0_m = dfs0.mean(axis=1)*v*Dt
dsm0_m = dfdsm0.mean(axis=1)
#====================================================================


# Plot all the persistence length changes and the mean
# Plot the persistence length with fitting, showing the Lp
fig, ax = plt.subplots(1,2, figsize=(20*cm,10*cm))
plt.subplots_adjust(wspace=0.3)
```

```python
def tenth(x):
  dftenth = int(round(0.3*(len(x)+.1),0))
  # 10% and round to the nearest whole
  return x[:dftenth]

for i in range(len(conff0)): # All trajectory plots
  ax[0].plot(tenth(s0_m), tenth(dfdsm0[i]), marker='o', \
  markersize=3, ls='--', lw=1, \
  color='lightblue', markerfacecolor='lime', label='_nolegend_')

ax[0].plot(tenth(s0_m), tenth(dsm0_m), marker='o', \
markersize=3, ls='--', lw=1, \
color='red', markerfacecolor='lime', label='Average') # Mean plot
#ax[0].set_yticks(np.arange(-1,1.1,0.25)) # For actin
ax[0].minorticks_on()
ax[0].tick_params('both', direction='in', top=True, right=True, \
length=8, which='major')
ax[0].tick_params('both', direction='in', length=4, which='minor')
ax[0].set_xlabel(r'$(S␣\cdot␣V␣\cdot␣\Delta␣t)\␣\mu␣m$', fontsize=14)
ax[0].set_ylabel(r'$\langle␣cos␣(\Delta␣\theta)␣\rangle_s$', fontsize=14)
ax[0].set_title('$r_{substrate}␣=␣%.2f$␣|␣$\Delta␣t␣=␣%.2f\
sec$'%(r,Dt), fontsize=14)

#--------------------------------------------------------
# Calculate the log of y and then do fitting
x = s0_m
y = dsm0_m


x = tenth(x); print(x)
y = tenth(y); print(y)

y = np.log(y); print(y)
df_xy = pd.DataFrame({'x':x, 'y':y})
df_xy = df_xy.dropna() # remove any NaN
x = np.array(df_xy['x']); print(x)
y = np.array(df_xy['y']); print(y)

ax[1].plot(x,y, marker='o', c='r', ls='--', lw=1, markerfacecolor='lime')

def func(x,Lp): # fitting function
  return 1*(-x/(2*Lp))
params, covs = curve_fit(func, x, y)
y = func(x,*params)
perr = np.sqrt(np.diag(covs[0])) # Error on params
#--------------------------------------------------------
ax[1].plot(x, y, label=r'Lp␣=␣%.4f␣$\pm$␣%.4f␣$\mu␣m$'\
%(params[0],perr)) # curve fit
#ax[1].set_yticks(np.arange(-3,0.1,0.5))
ax[1].minorticks_on()
ax[1].tick_params('both', direction='in', top=True, right=True, \
length=8, which='major')
ax[1].tick_params('both', direction='in', length=4, which='minor')
ax[1].set_xlabel(r'$(S␣\cdot␣V␣\cdot␣\Delta␣t)\␣\mu␣m$', fontsize=14)
ax[1].set_ylabel(r'$log␣\langle␣cos␣(\Delta␣\theta)␣\rangle_s$', \
fontsize=14)
ax[1].set_title('$r_{substrate}␣=␣%.2f$␣|␣$\Delta␣t␣=␣%.2f\
sec$'%(r,Dt), fontsize=14)

ax[1].legend()
plt.savefig('fig/LpF-'+str(round(r,2))+'-Dt'+str(round(Dt,2))\
+nm+'.pdf', format='pdf', bbox_inches='tight')
plt.show()
#===================================================================
```

You may fork the code in B.1 on Github.

B.2 Myosin Lifetime Calculation

```python
#!/usr/bin/env python
# coding: utf-8
# Input:  motor states
# Output:  lifetime data
#==============================================================
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from ast import literal_eval
import glob

#==============================================================
files = glob.glob('MotorStates_Ts0.**.txt')
files = sorted(files, key=lambda x:x[-20:])

#==============================================================
columns = ['ts','im','mt','c','xc','yc','zc','xm','ym','zm',\
'fx','fy','fz']
# ts = timestep, im = motor index, mt = motor type (active = 1,\
# defective = 2)
# c = contact state, xc|yc|zc = binding motor head position,\
# xm|ym|zm = binding motor root position
# fx|fy|fz = xyz force experienced by motor

for r in files:
  print(r)
  ms = pd.read_csv(r, names=columns, delim_whitespace=True)

  #==============================================================
  # Separate active motor and defective binding motors.
  ms_act = ms[ms.mt == 1]
  ms_act = ms_act.reset_index(drop=True)
  ms_def = ms[ms.mt == 2]
  ms_def = ms_def.reset_index(drop=True)

  #==============================================================
  # Lifetime metric: during one lifetime, a binding motor, ( xm,ym)
  # must retain index 'im'
  # and also contact state, 'c' in the next immediate time step 'ts'
  #act_xy = np.around( ms_act[['im','c','xm','ym']], 6).values.tolist()
  #def_xy = np.around( ms_def[['im','c','xm','ym']], 6).values.tolist()

  ms_act = ms_act.drop(['ts','mt','xc','yc','zc','zm','fx','fy',\
  'fz'],1) # remove the unused columns
  ms_act = ms_act.groupby(ms_act.columns.tolist(), \
  as_index=False).size() # count duplicated rows
  ms_act.rename(columns={'size':'lyf'}, inplace=True)

  ms_def = ms_def.drop(['ts','mt','xc','yc','zc','zm','fx','fy',\
  'fz'],1)
  ms_def = ms_def.groupby(ms_def.columns.tolist(), \
  as_index=False).size()
  ms_def.rename(columns={'size':'lyf'}, inplace=True)

  #==============================================================
  # columns will be: |im|c|xm|ym|life| *
  ms_act.to_csv(r[12:-4]+'act_with_lyf.csv', header=False, \
  index=False, float_format='%.6f') # idx,contact,x,y,life
  #--------------------------------------------------------------
  try:
    ms_def.to_csv(r[12:-4]+'def_with_lyf.csv', header=False, \
    index=False, float_format='%.6f') # idx,contact,x,y,life
  except:
    print('Saving␣m2␣passed:␣'+r) # No defective motors in R =1.0
```

Other programs are available on Github [2018/19] [2020] [2021].

C Published Works

The following are related presentations and publications that were done before the final submission of this dissertation:

1. **“The Impact of Defective Motors on Biosensor Integrated with Actin Filaments and Myosin“**
   Samuel Macharia Kang'iri and Takahiro Nitta
   *PRESENTATION [20189G]: BSJ2020, **Biophysical Society of Japan**, 2020*
   [https://doi.org/10.2142/biophys.60.S1]

2. **“Effect of defective motor proteins on the performance of biosensors integrated with motorproteins“**
   Samuel Macharia Kang'iri, Dan V. Nicolau, and Takahiro Nitta
   *PRESENTATION [P2.116]: BIOSENSORS2021, **Elsevier** - Biosensors and Bioelectronics*, 2021
   [https://app.oxfordabstracts.com/events/1657/program-app/session/12709]

3. **“A Mathematical Model Predicting Gliding Speed of Actin Molecular Shuttles over Myosin Motors in the Presence of Defective Motors“**
   Samuel Macharia Kang'iri and Takahiro Nitta
   *BOOK CHAPTER/PAPER: **Springer** - BICT*, 2021
   [https://doi.org/10.1007/978-3-030-92163-7_17]

4. **“Simulations of Actomyosin-Based Molecular Shuttles Controlled by External Force“**
   Samuel Macharia Kang'iri, Andrew Salem, Dan V. Nicolau, and Takahiro Nitta
   *PAPER: **IEEE** - MHS*, 2021
   [https://doi.org/10.1109/MHS53471.2021.9767188]

5. **“Effects of defective motors on the active transport in biosensors powered by biomolecular motors“**
   Samuel Macharia Kang'iri, Andrew Salem, Dan V. Nicolau, and Takahiro Nitta
   *PAPER: **Elsevier** - Biosensors and Bioelectronics*, 2022
   [https://doi.org/10.1016/j.bios.2022.114011]

6. **“Motility resilience of molecular shuttles against defective motors“**
   Samuel Macharia Kang'iri and Takahiro Nitta
   *PAPER: **IEEE** - Transaction on Nanobioscience*, 2022
   [https://doi.org/10.1109/TNB.2022.3170562]

7. **“Linking path and filament persistence lengths of microtubules gliding over kinesin“**
   May Sweet, Samuel Macharia Kang'iri, and Takahiro Nitta
   *PAPER: **Nature** - Scientific Reports*, 2022
   [https://doi.org/10.1038/s41598-022-06941-x]

D Recollections

Figure D.1: Supervisors and I (2019). **(a)** In Nagoya Castle Japan. **Left:** Prof. Dan V. Nicolau from McGill University, Canada. **Middle:** Sam Macharia from Dedan Kimathi University of Technology, Kenya. **Right:** Prof. Takahiro Nitta from Gifu University, Japan. **(b)** In Gusto Restaurant Gifu Japan. **Left:** Prof. Minoru Sasaki from Gifu University, Japan. **Right:** Sam Macharia from Dedan Kimathi University of Technology, Kenya.

Samuel Macharia Kang’ iri (サミュエル マシャリア カンギリ)：論文筆頭著者
岐阜大学大学院工学研究科

新田高洋准教授（左）、Samuel Macharia Kang’ iri さん（右）

Figure D.2: Research work (2020, 2021). **(a)** A typical lab day in Prof. Nitta Lab, Gifu University, Japan, here doing data analysis. **(b)** A press release in Gifu University [https://www.gifu-u.ac.jp/news/research/20220207.pdf] after a publication [63].

Figure D.3: A photograph after the defence presentation exam (2022).

INDEX

The End