ABSTRACT

Title of Dissertation: FUNDAMENTALS OF EMBODIED REPRESENTATION: ROBOTICS WITHOUT A RULER

Levi Souza Burner
Doctor of Philosophy, 2025

Dissertation Directed by: Professor Yiannis Aloimonos
Department of Electrical and Computer Engineering

Imagine sitting at your desk, looking at various objects on it. While you do not know their exact distances from your eye in meters, you can reach out and touch them. Instead of an externally defined unit, your sense of distance is inherently tied to your action's effect on your embodiment. Animals ranging from insects to humans use such a concept of distance to determine what actions to take. They must because most animals do not know what an external scale, such as the meter, is. Instead, they must make do with an internal unit, or embodied unit, that is somehow measured using the signals available from the body. In contrast, today's robots almost exclusively rely on the meter. Their bodies are measured in meters, their sensors are calibrated to the meter, and consequently, their control, planning, and vision systems use the meter. Further, extensive effort is put into calibrating these systems end ensuring those calibrations do not degrade over time, else the robot will stop working.

If robots could represent the world using their own sense of distance, they would not re-

quire such calibration or precise engineering. Instead, they could use uncalibrated sensors, with uncalibrated bodies, to accomplish tasks such as clearing obstacles, jumping gaps, and manipulating objects. Inspired by this problem, this dissertation develops a visuomotor approach through which robots can accomplish such tasks without prior knowledge of an external unit. The key to the approach is using a system's own actions, or control inputs, as internal feedback from which a unit is implied and used to estimate quantities such as the body's size, motor dynamics, or position of objects in the world. The resulting techniques are called "Embodied Representation" because they consist of measurements in terms of the signals available to the robot's body itself, without calibration to an external scale.

The development of Embodied Representation is detailed in this dissertation, resulting in techniques and algorithms for fundamental problems encountered by embodied systems. First, a specific method for using time-to-contact, a bio-inspired visual representation, with acceleration, is used to achieve stable closed-loop control even when the units are embodied and thus unknown. Subsequently, a general framework "Embodied Visuomotor Representation" is developed for estimating and using such representations. The resulting algorithms for uncalibrated clearing and jumping mirror natural behaviors observed in bees and gerbils. Next, the use of internal feedback for manipulation is studied, specifically for key insertion. The robot compares its own wiggling signal to tactile feedback in order to guide an insertion process. The result can insert keys into four types of locks. Furthermore, in an assembly benchmark, it outperforms a reinforcement learning baseline trained on the objects. Finally, the impact of vibration on visual perception, as induced by limbed or winged locomotion, is studied. This results in a specialized video stabilization algorithm, which represents the world in a coordinate frame different from the system's body and stabilizes video from a tailless ornithopter known for its aggressive shaking.

FUNDAMENTALS OF EMBODIED REPRESENTATION: ROBOTICS WITHOUT A RULER

by

Levi Souza Burner

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2025

Advisory Committee:
Professor Yiannis Aloimonos, Chair/Advisor
Dr. Cornelia Fermüller
Professor Perinkulam S. Krishnaprasad
Professor Dinesh Manocha
Professor Guido de Croon
Professor Nikhil Chopra, Dean's Representative

To Santina and my family.

Acknowledgments

I do not know how to express the gratitude due to the many who have supported my PhD studies. It is simply overwhelming. This journey could not have been completed without you.

Early on, it was my advisor Prof. Yiannis Aloimonos, who would come to the lab and ask "Why must we teach robots what a meter is?". It took a few years for me to understand what this question meant, and longer still to find a meaningful answer. Throughout this period, he taught me to question what seems surely established, communicate concepts that almost defy language, discriminate the significant from the insignificant, and appreciate that in its highest form, science is a grand endeavour, spanning millennia, with no regard for disciplinary boundaries. My development would not have been possible without Dr. Cornelia Fermüller, who constantly encouraged me to interpret mathematical formulations carefully and through the lens of the biological sciences, in a way that made their properties clear. I will never forget the extreme patience and personal kindness Yiannis and Cornelia showed me over these years. It is an ideal I will strive to uphold in my career. I cannot thank you enough.

There are a few professors who have supplemented my training in exceptional ways that I must thank. Prof. P. S. Krishnaprasad taught me to use mathematics not just as a formalism, but as a way to understand the phenomena we observe around us. Prof. Guido de Croon's work originally inspired me to study bio-inspiration and bio-mimicry in robotics, and in multiple instances made clear, specific suggestions with transformative effects on my thesis. Prof. André Tits

taught me a healthy dose of optimal control and provided support to attend my first conference. But, more importantly, he taught me the importance of attention to detail and fair consideration while I was his Editorial Assistant for the Rapid Publications of Automatica. Finally, Prof. Gil Blankenship taught me to teach students to accomplish the impossible by not telling them that what they're trying to do is impossible. Thank you.

I must thank many peers, most of whom are also labmates in the PRG Lab. Their support and camaraderie cannot be understated or taken for granted. Your dedication to knowledge, astounding creativity, and willingness to roll up your sleeves will no doubt have transformative effects on the world. I have been amazed by all of you and learned so much from your suggestions and deep conversations. Nitin Sanket and Chahat Deep Singh whose work attracted me to Maryland in the first place and then served as mentors. Xiaomin Lin, who introduced me to the lab. Matthew Evanusa, Snehesh Shrestha, Kanishka Ganguly, Chethan M. Parameshwara, Anton Mitrokhin, Eadom Dessalene, Vaishnavi Patil, Angelos Mavrogiannis, Zhiyuan Hua, Dehao Yuan, Jingxi Chen, Amir-Hossein Shahidzadeh, Botao He, Pavan Mantripragada, Gabriele Caddeo, Yianni Karabatis, Hirad Yazdankhah, Jiayi Wu, Naitri Rajyaguru, Matthew Jacobsen, Junaid Merchant, Daniel Callow, Kevin Zakka, and Yuval Tassa.

I am indebted to several talented master's and undergraduate students who helped me in my research and allowed me to learn to mentor. Sakshi Kakde, Chenqi Zhu, Stephen Jardim, Jason Zhao, Jack Mirenzi, and Rohit Kommuru.

The support of the Maryland Robotics Center in my studies has been critical. In addition to financial support, it has provided shared facilities, equipment, and teaching experiences that made my work possible. I must in particular, thank Ivan Penskiy; his extreme care and diligence allowed the facilities I depended on to be developed and maintained.

My family was incredibly supportive of this journey. Without them, I don't know how I would have managed through difficult periods. You never doubted me, and I am blessed to have all of you. In particular, my wife Santina supported me throughout the entire journey while pursuing her own strenuous studies. Somehow, we managed to get married in the middle of everything. She alone knows how difficult this journey has been, and I don't know how I would have completed it without her. Finally, I must thank Ellie, the sweetest dog in the world.

I would like to acknowledge financial support from the Graduate School's Dean's Fellowship, Computation and Mathematics for Biological Networks Fellowship, Maryland Robotics Center Graduate Research Assistantship, Ann G. Wylie Fellowship, the Future Faculty Fellowship, USDA, and the NSF, which gave me the freedom to write such a thesis.

I sincerely hope I have not left anyone out. If I did, I apologize from the depths of my heart.

Thank you all!

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

Mammals walk, fish swim, and bees fly while relying on visual perception as feedback to drive muscular responses. Further, while their morphologies are strikingly different, they all have the same basic abilities, such as navigating, avoiding obstacles, manipulating objects, and chasing targets. Surprisingly, they do this despite widely varying computational systems. Mammals can easily recognize objects and decompose scenes. Meanwhile, fish do not even have a visual cortex, and bees have 1/10th the neurons of a fish. Further, they can even learn to control themselves quickly; a giraffe learns to walk within an hour of birth. This situation begs the question: is there a principle that can realize visuomotor control in a broad variety of systems?

This question has been asked for a long time, but until recently, theories could only be developed through psychophysics and animal experiments. Further, due to their nature, both fields rarely admit the detailed computational explanations needed to guide an implementation. However, a relatively new field, robotics, and its associated mathematics for visual perception, control, and dynamics, offers a new path. With these tools, we can propose mathematics and compare its properties to the vision-based control demonstrated by animals. By developing the most closely aligned frameworks, we may realize machines with robust, adaptive visuomotor abilities like animals.

A literature review reveals an immediate impediment. Robot's sensors, physical forms,

and controllers are widely assumed to be pre-calibrated to a unit like the meter[1]. However, most animals do not know what a meter is. This dissertation questions this unnatural assumption and develops mathematical models inspired by the uncalibrated condition of an animal. The resulting concept, "Embodied Representation", eschews the meter for self-calibration in an implicitly defined, internal unit, provided by all motor systems. That is, the signals sent to actuators that themselves exert forces are used to represent precisely "where" and "how big" the robot and world are.

This process of reusing the signals sent to the actuators is called internal feedback and, as will be discussed, has broad support in the neuroscience literature. In what follows, we discuss the historical development of this concept, the de-facto standard approaches to robot perception, and bio-inspired alternatives that fall inline with neuroscientific findings. Finally, we outline four fundamental problems in robotics that would be substantially easier if internal feedback is used to form embodied representations that drive actions. Their solutions in the subsequent chapters suggest that Embodied Representation is a paradigm shift that will lead to robust, ubiquitous robots that are easy to manufacture and deploy in a broad variety of applications.

The chapter concludes with the dissertation's research objectives and organization.

1.1 Prior Work on Internal Representations

In the middle of the 19th century, Hermann von Helmholtz wrote the pioneering work in the physiology of visual perception, "Treatise on Physiological Optics" [3]. In it, he detailed basic models for phenomena ranging from geometric projection to depth perception and accompanied

[1]While the word "meter" will be used extensively in this thesis, all occurrences could be changed to "externally defined unit", meaning any arbitrarily defined length such as micrometers, yards, or miles.

them with empirical experiments. Helmholtz was concerned primarily with the basic physiology and physics behind visual perception that could be described with exact mathematical laws and studied with precise empirical tests. This led to a robust science of vision, but left something to be desired in understanding perception. That is, the methods biological systems use to turn images falling onto their retina into the actions they must take to survive.

As a result, a framework for understanding perception was proposed by Jakob Johann von Uexküll in the early 20th century. In his work "Theoretical Biology", Uexküll most famously describes the concept of "Umwelt", that is the perceptual world of an animal as dictated by the senses afforded to its own body [1]. The famous canonical example is a tick, whose world consists of the smell of skin-oils, temperature, and touch. Consequently, it is likely that the ticks have no understanding of visual perception and may not even represent the world in a 3D manner.

Today, Umwelt carries on as Uexküll's most famous contribution. However, Uexküll introduced another important concept. He proposed that internal feedback exists between the brain's motor system and perceptual system, which he called the Werk-organ and Merk-organ, respectively. Uexküll assumed that this internal feedback exists in all of the higher animals (such as mammals), but not the lower ones (such as sea anemones). The lower animals, he claims, are limited to simple stimulus-response behaviours, where an event in the world, such as light pattern, contact, or vibration, lead immediately to a single pre-determined motor response. However, higher animals, and in particular humans, have the ability to use tools, open doors, and generally recognize the utility of objects for purposes such as shelter, play, hunting, and gathering resources. To explain this ability, Uexküll suggested that the internal feedback from the Werk-organ to the Merk-organ allows the perceptual system to upgrade the objects recognized from the visual system from mere objects to implements. That is, objects which are useful for a certain

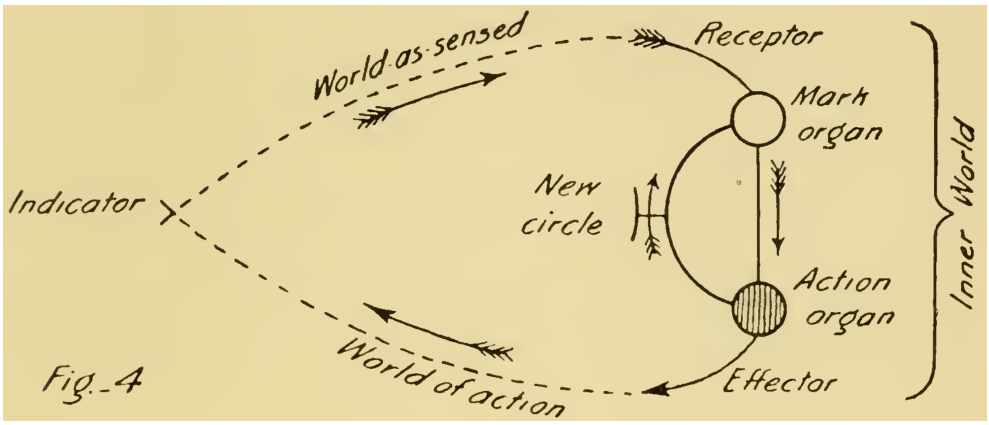

Figure 1.1: Uexkull's "function circle" reproduced from the 1926 English translation of "Theoretical Biology" [1]. In the english translation the "Werk-organ" is translated to "Action-organ". Uexkull's work considered the concept of internal feedback going from the motor system to the perceptual system, called the "New circle", and ascribed it to only the higher animals.

purpose, as may be needed by the motor system that aims to accomplish a task.

In Figure 1.1, a figure from Uexküll's "Theoretical Biology" is reproduced. In it, we find the Merk-organ (perceptual system) and Werk-organ (motor system) that were postulated to be contained in an animal's embodiment. They interact with the world in a loop, where the actions of the system effect the world, and the world effects the responses of the perceptual system. Uexküll called this the "function circle" and proposed that all biological systems adhere to it, not just those with eyes.

Uexküll recognized that all animals must use their own units to represent magnitudes. Because this fact is somewhat obvious, Uexküll went even further. He questioned where we get our sense of direction, in particular the 3 cardinal directions. In Chapter 1 of "Theoretical Biology", he concludes that our sense of direction must come from senses capable of responding to each direction. For example, suppose we lived in a 3D world but were restricted to moving in a plane with eyes that only see horizontally out onto that plane. Then, Uexküll posits, our Umwelt would only contain a two dimensional world. The third dimension would be unobservable to

us. Further, Uexküll pointed out that our sense of orthogonality, that is one direction between independent on each other, is similarly artificial and must arise from the pattern of receptors on our sensory apparatus.

However, directions don't have magnitude, which is critical for almost all behaviors. Uexküll addresses this by discussing where our sense of time comes from, assuming that their is a discrete, small unit of time that each system perceives as the smallest possible increment of time, that from multiples of this increment we gain the concept of magnitude. Subsequently, this conception of magnitude is used to measure distances along directions.

The contribution of Uxeküll most relevant to this dissertation is found on pages 66 and 67, Chapter 2, of "Theoretical Biology". Here, it is pointed out that while a system's inner units can be measured by conventional ones, such as the meter, such notions of distance are fundamentally different. One is due to our physical nature, arising from our Umwelt, dictated by embodiment. It is fundamental and inescapable that each system has this internal (inner) unit and all systems must experience distance in similar ways despite their units have different absolute magnitudes. Uxeküll claims that his work is the first time this concept was expressed.

In summary, with Uxeküll, we find two concepts that are integral to Embodied Representation. One is internal feedback from the motor system to the perceptual system, and the second is the idea that all biological systems have their own units, arising in their Umwelt, that are somehow derivative of their physical embodiment. However, Uxeküll stopped short of determining how this might be done and did not explicitly connect the concept of internal feedback to his conceptualization of inner units.

Following Helmholtz and Uxeküll, the field of neuroscience attempts to blend the low-level physical ideas in the Helmholtz approach to the higher level, conceptual approach of Uxeküll. A

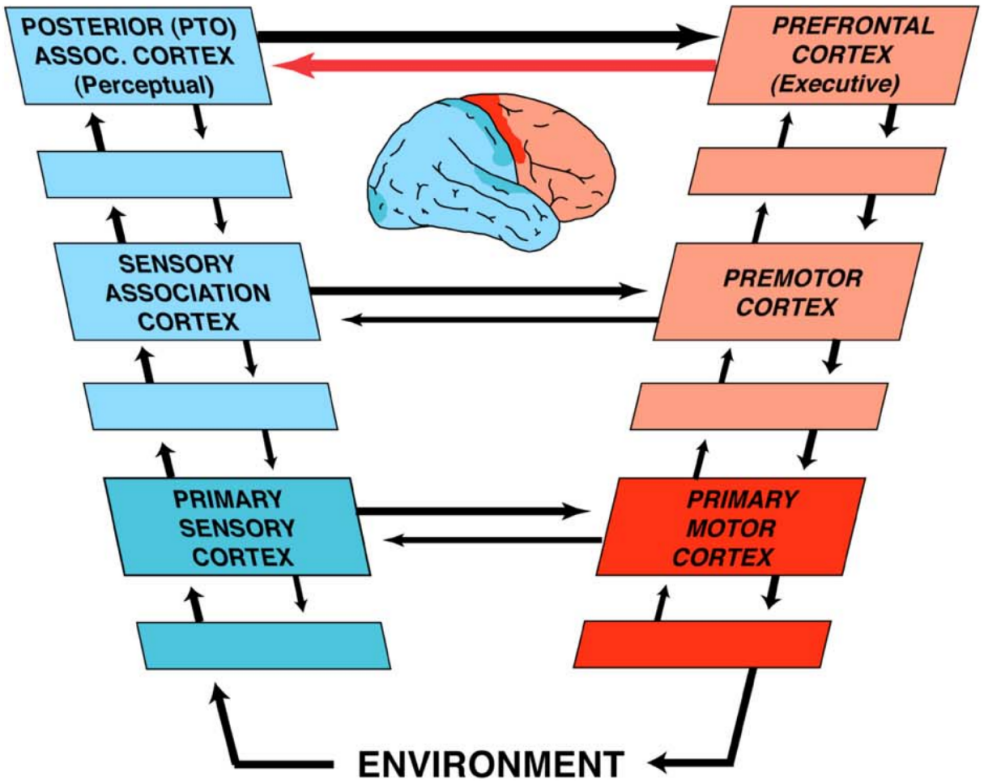

Figure 1.2: Fuster's hierarchy reproduced from [2] © 2014 IEEE, reproduced with permission. Fuster describes a stack of processing units going from perception or low level motor outputs to high level cognition. Internal feedback goes from each motor layer to the corresponding perceptual layer. The role of internal feedback is described as aiding prediction in a general sense.

central figure in this development is Joaquin Fuster who championed an extension and reinterpretation of Uxeküll's inner feedback loops [2]. Often referred to as Fuster's hierarchy, we reproduce it's characteristic illustration in Figure 1.2. From Fuster, a more detailed architecture for perceptual processing emerges. Instead of a single perception and motor system, there is a hierarchy, a stack, of processing units, going from the sense and motor outputs and ending in the cognitive domain (that is, high-level decision making). Internal feedback is posited as going from each processing stage in the motor system to a perceptual processing stage at the same level in the perceptual hierarchy.

In Fuster's hierarchy, the highest level of processing is the cognitive domain. Fuster claims

that the internal feedback at this level most closely corresponds to the internal feedback in Uxeküll's framework that goes from mark-organ to welt-organ. Indeed, Uxeküll's specifically claims that this feedback is to guide the perceptual system to recognize objects in the perceptual system to recognize them as implements, and only the higher animals are capable of this. But then, what is the role of motoric internal feedback at the other levels?

Fuster proposes that its role is prediction, and spent a lifetime developing neuroscientific studies showing the predictive power of the prefrontal cortex. Prediction can take many forms, but the most relevant to Embodied Representation is the idea that an animal must predict what it will perceive after it takes an action. When vision is accounted for, it becomes clear that this requires an understanding of distance or an assumption about the operating conditions (this is discussed in detail in Chapter 3 of this dissertation). But then, where does this sense of distance come from exactly? It must be related to Uxeküll's conception of internal units, but it does not arise from his conception of internal feedback.

An answer to these questions is the main concept in this dissertation. That is, "Embodied Representation", a way for systems (such as robots) to estimate distances, their shape, and perform tasks using their inner, or embodied units, that arise from solving a prediction problem that depends on their embodiment's internal feedback signals.

1.2 Perceptual Representations in Robotics

The study of artificial visual perception ostensibly begins with David Marr's work titled "Vision" which was mostly written in 1979 and published posthumously in 1982 [4]. Tragically, "Vision" was written in response to Marr's diagnosis with terminal illness, which he succumbed

to at the age of 35. “Vision” contains the first comprehensive computational framework for computer vision. The work is broad, proposing multiple frameworks for organizing the study of visual computation. They range from the physical (similar to Helmholtz) to the purely conceptual (similar to Uexkül). The middle layer is algorithmic which requires writing down specific algorithms that do specific tasks without committing to a particular computational medium or physical implementation.

Marr’s paradigm for vision envisions a camera as a passive instrument. It is carried by a machine that moves, but what it sees is an accidental consequence of where the camera is mounted on that machine. Subsequently, it was proposed that algorithms would reconstruct the world in 3D around the robot and localize it within that representation. Such an approach, if it works, is always sufficient because, as is well established by the graphics community, 3D models and the position of the camera are enough to reconstruct an image. Thus, such things are estimated, all possible information out of the image, or so the theory goes.

Because of Marr’s early passing, it can only be assumed that his account was incomplete. Indeed, biological systems engage in constant and purposive movement in order to acquire information about the world and better understand what they are looking at. Further, it is unclear if biological systems necessarily represent the world in 3D; it is possible that many simply extract the least information required to accomplish a task.

This situation resulted in new frameworks for vision that consider the cameras as a moving agent. In 1988, Yiannis Aloimonos published “Active Vision”, which considered computer vision problems under the assumption of known or partially known motion (or action) [5]. The resulting computer vision problems have additional constraints, resulting from the known motion, that make ill-posed problems well-posed. Examples include shape from shading, contour, and the

general structure from motion problem. In the same year, Ruzena Bajcsy published "Active Perception" which considers more general problems and results in a hierarchy of activeness ranging from controlling the camera's apparatus to controlling the semantic interpretation of what a system perceives [6]. The hierarchy of activeness in "Active Perception" is reminiscent of Fuster's hierarchy, though the connection is not made explicitly. Later, in 1991, Dana H. Ballard published "Animate Vision", which considers a form of purposeful motion arising from a cognitive interpretation [7, 8]. Here, the world is viewed as a sort of turing tape, that a system lives within, and sequentially queries the world, by looking act a particular object or point, to determine what to do.

As robotics advanced through the late 90's, 2000's, and the present. Marr's paradigm, which presents vision as a passive problem of reconstruction, gained popularity. Computational tools such as SIFT allowed tracking visually salient points across many frames [9]. SLAM algorithms [10,11] gained popularity which could solve the structure from motion problem incrementally, as a robot moves through the world, and fast planning algorithms such as RRT [12] enabled calculation of routes through the resulting maps. Today, it can be assumed that commercial systems that use vision based state estimation, such as drones, virtual or augmented reality headsets, and cars, are using tools based in these frameworks.

While the state-of-the-art in robotics developed through the lens of passive reconstruction, the active approaches suggested by Aloimonos, Bajcsy, and Ballard continued to be studied through the lens of bio-inspiration. That is creating systems that perceive like animals, not because we need to, but because we want to understand the animals themselves. Work on these lines often focuses on the low-level visual representations that might be computed by biological visual systems and how they can be used to effect actions. A principal quantity in this field is time-to-

contact (TTC). It rose to fame in 1976 when David Lee proposed it might be easily measured by the human visual system, and is sufficient to allow humans to keep cars from coming into contact with each other while braking [13]. Since then, numerous studies in robotics have attempted to use time-to-contact for visual control, and its measurement is still actively studied. Consequently, this dissertation focuses on time-to-contact as a fundamental perceptual representation, resulting in a detailed mathematical examination of its properties in Chapter 2.

1.3 What are the Units?

In all the above perceptual frameworks, a method for estimating distance in internal units is not explicitly considered. Instead, if vision alone is used, the world's representation is assumed to be to scale (such as when considering monocular structure-from-motion). Then, it is assumed that these representations will be scaled to the meter, for example using an externally calibrated stereo baseline, LIDAR, or an IMU. To affect action in external units, the robot's body must also be calibrated to the meter. While this methodology is sufficient, it is well known that today's robots are categorically expensive, available in only a few physical configurations, and must be precisely calibrated, else they do not work.

This situation is striking. Does a dog know what a meter is? Does it ever go through a calibration process involving precisely made checkerboards or mechanical fixtures that place limbs in known configurations? If it grows, does it need help from its owner to recalibrate so that it can walk again? Of course, the answer is a resounding "NO!".

Thus, it seems prudent to study systems capable of representing themselves, in their own units, due to signals from their embodiment. Next, we examine a few activities that are easy for

animals but difficult for robots where the success of animals may be attributed to their use of internal units that comprise an Embodied Representation.

1.3.1 Clearing

Almost all animals can clear obstacles. That is, they know what actions to take so that they do not make contact with a perceived structure. While robots have been able to imitate this behavior in various ways for a long time, commercially available systems grossly depend on calibrated 3D self-models and sensors with which they determine if the perceived object is further away than the body. If the robot's body changes, due to damage or attachment of additional hardware, the model must be updated by a human. Further, if the sensor's accuracy drifts due to temperature, damage, or mechanical tolerances loosening over time, it must be recalibrated, again by a human, so that the measurement's units are congruent with its 3D self-model.

This situation is far from the biological case. Many animals change in size substantially during their lifetime, and not once do they need to be manually calibrated by another system. Further, most animals do not explicitly know the parameters of their vision-based perceptual system, such as inter-pupilary distance, the length of their limbs, or the strength of their muscles. Further, while some animals have methods for estimating absolute distances to nearby objects (such as the echo-location system of the bat), the mapping from stimulus to neural response to a correct action is still somehow developed by the animal itself. Thus, it seems likely that some type of internal units must be in play with which an animal can measure if it is too close, or too far, from an obstacle to be cleared. If so, then artificial systems using such units can be expected to maintain their clearing abilities in the real world, over long time periods. This is because

the usage of internal units, avoids the intermediate information, which can only be supplied by humans, that transforms perceptual representations and models of the body to external units like the meter.

But where precisely do these internal units come from? It has been established that even bees understand the size of objects relative to their own body size. Further, they can assess this ratio before coming close to an opening [14]. This means their own bodies have not yet physically interacted with the opening, and we so the ratio between the body size and the opening size is unobservable from such a scenario alone. Thus, the opening must be measured in some other unit, the bodies size is also known in that unit, and consequently the ratio can be determined.

In Chapter 3 we show how this situation can be resolved using units of action, which can be estimated by the internal feedback suggested by Uxeküll and Fuster. The result is a real robot that can estimate its own size using its own acceleration signals and visual inputs. Further, it can estimate the size of openings in that same unit. The process takes just a few seconds, demonstrating that such representations are easy to compute and can be estimated online.

1.3.2 Jumping

When animals jump over an object or a gap, they must jump the correct distance to minimize the risk of injury. Indeed, dogs regularly participate in agility competitions, which require precise maneuvers. Gerbils can jump gaps of varying size and, notably, appear to do "test oscillations" prior to the jump which vary in frequency and magnitude depending on the distance they must jump [15]. As in the clearing example, if any of these systems grow or change their size, they can quickly learn to adjust their jumping behavior to still accomplish the task. But more im-

portantly, they must also have an understanding of their own muscular dynamics, because there is little time to correct for errors in the midst of executing a jump. Instead, what one must do to execute the jump has to be estimated online, as a mostly open-loop input to the embodiment. Once again, the question "What are the units?" appears.

Today's robots can jump gaps, and some of the demonstrations are even spectacular. However, all existing examples require extensive calibration and often existing kino-dynamic models that are subsequently used to test or train task specific behaviors. If the robot is changed, the designed policies will no longer work.

In Chapter 3, we show how this situation can also be resolved by using units of action. Here, however, the incorporation of muscular dynamics results in a surprising mathematical development that there are multiple choices of internal unit, none necessarily better than the other. For pragmatic purposes, one particular unit is chosen, resulting in a simulated agent that can jump a gap of unknown size without knowledge of its own strength, the strength of gravity, or precise knowledge of its muscular dynamics. The process of estimating the required Embodied Representation takes a few seconds and imitates Gerbil behavior.

1.3.3 Manipulation

Finally, we consider manipulation, the ability of a system to pick up, move, insert, or change the physical configuration of its world through contact. This problem lies at the heart of robotics, but despite decades of research and the rather extreme economic potential of robot's that can manipulate objects, there are no commercial systems that can handle a wide array of problems automatically. Further, in manufacturing contexts, human engineers painstakingly preprogram

specific behaviors for specific machines. Finally, while some success has been realized in robotic manipulation using reinforcement learning over a wide range of embodiments and simulated problems, no system has yet achieved widespread success.

In the previously discussed problems of clearing, jumping, and flapping flight, it was suggested that the system's own actions could provide the unit with which to measure distances and effect control. In those cases, the emphasis was on estimating Euclidean distances in a three dimensional space. However, manipulation, while not necessarily in a Euclidean space, and whose dynamics cannot be easily described by integration of acceleration, is no different. Consider a robot that must insert a key into a lock. It knows the actions it took, and it can feel, or see, how far the key is inserted into the lock. By adjusting the pose of its hand that grasps the key, it attempts to insert the key further. What forces should it exert? What then are the units of the estimates it uses to determine those forces?

For this problem, the contact dynamics cannot be modeled by a simple peg-in-hole model, where the objects have smooth surfaces. Further, we cannot expect to learn a general strategy offline. Instead, somehow, the robot must learn how its own actions affect the behavior of a manipulated object to adjust its behavior and accomplish the goal. In Chapter 4 we consider this problem and show that internal feedback, resulting in a necessarily Embodied Representation of how a future action will affect the manipulation process, results in a single control law that can insert keys into a variety of locks. Further, it outperforms a reinforcement learning baseline on an assembly benchmark, where the learned baseline was trained on the objects.

1.3.4 Vibration Compensation

Robots and animals that rely on limb or wing based motion shake. While many methods exist in nature to solve this problem, such as active mechanical stabilization by the neck, similar systems are currently difficult to replicate in robots. Interestingly, animals with foveated vision systems, such as humans, constantly experience small rapid eye movements. However, their perception of the world appears to be that of a stable viewpoint. Inspired by this, we develop a specialized video stabilization algorithm, that can achieve "Artificial Microsaccade Suppression". That is, removing small rotation like movements from an aggressively rotating body. We demonstrate it on a tailless flapping robot, which previously proved exceptionally difficult to use with vision based sensing. The result is the first stabilized videos from the platform, which are amenable to downstream, real-time applications. Fundamentally, the method works by representing visual inputs in a coordinate frame separated from the body. That is, we find an appropriate representation for a tailless flapping robot, that compensates for its embodiment's motor system, and results in a usable, real-time, computable video feed.

1.4 Objectives

The above questions suggest that robots should use their own units and convenient coordinate frames for control in problems like clearing, jumping, manipulating, and flapping based flight. Further, it is expected that internal feedback, from motoric to sensing systems will be critical to realizing artificial systems that are capable of estimating distances, and choosing actions using their own units. This dissertation examines these problems from a classical perspective, deriving the basic properties we can expect from these assumptions and culminating in a successful

demonstration of each of the associated problems.

Thus, we establish the theoretical foundations of embodied representation, which will hopefully lead to a rich vein of future research and practical deployments of robots that understand the world in their chosen unit and coordinate system, so that they can adapt to changing environments and embodiments, without requiring precise manufacture or design, much like animals.

1.5 Organization

Chapter 1 (the current chapter), motivates the dissertation's focus and chosen approach. It begins with Helmholtz foundational work on perception in the 19th century, focuses on the conceptualization of internal feedback developed by Uxeküll and Fuster in the 20th century, and ends with a discussion of problems in robotics that should be much easier to solve than they currently are. Embodied Representations, due to internal units, are then proposed as a potential solution.

Chapter 2 establishes preliminaries for a bio-inspired approach to visual representation. It considers time-to-contact, a quantity thought to be the basis for biological perception from insects to humans. Several mathematical results are developed showing that time-to-contact over time results in a trajectory to scale. Further, this can be generalized to 3D motions instead of motion restricted to the optical axis. Acceleration is incorporated as an analog of action, and it is shown that accurate distances to tracked objects can be measured from this methodology. Finally, the first indication that using internal feedback as a source of units can have desirable properties is briefly examined. That is, a robot without knowledge of its strength can maintain a stable closed

loop despite artificial variation of its strength by a factor of 2. Meanwhile, a traditional approach based on the meter becomes unstable.

Chapter 3 carefully considers the usage of internal feedback for establishing an embodied unit. It contrasts the approach to a large body of existing work, including Visual Inertial Odometry, Visual Servoing, Direct Optical Flow Regulation, Tau Theory, and Glenberg's theory of memory through actions. Additionally, it greatly expands on the mathematics that were only briefly touched on in Chapter 2, resulting in a general and classical formulation for achieving Embodied Representation. Further, two algorithms are presented for realizing uncalibrated clearing and jumping. Here, systems without knowledge of their size, strength, or any distance in the world become capable of clearing obstacles of unknown size and jumping gaps of unknown width in just a few seconds.

Chapter 4 considers the usage of internal feedback for a manipulation problem, key insertion. While the robot in question uses the meter (it is a calibrated arm), the perception and the relation between action and perception are uncalibrated due to the unknown contact dynamics between the key and lock. Subsequently, it is demonstrated that internal feedback from the actions to the tactile perceptions results in a single algorithm that can insert four keys into four types of locks without any further parameter tuning. Additionally, when tested on an object assembly benchmark, that same algorithm and parameters achieve a 98% success rate while a reinforcement learning baseline, which was trained on the objects, achieves only 86%.

Chapter 5 considers the problem of vibrations induced by limb or wing based motion on the body. As an extreme example, tailless flapping robots have so far eluded use of vision based sensing because of their aggressive vibrations induced by their wings. In response, a specialized video stabilization algorithm is developed that does not rely on feature matching and is inspired

by our inability to perceive the microsaccades of our own eyes. The result achieves the first stabilization of the video of a tailless flapping robot and is free from apparent distortion and runs in real time. The result opens up many future directions in flapping robot research.

Chapter 2: TTCDist: Fast Distance Estimation From an Active Monocular Camera Using Time-to-Contact

Distance estimation from vision is fundamental for a myriad of robotic applications such as navigation, manipulation, and planning. Inspired by the mammal's visual system, which gazes at specific objects, we develop two novel constraints relating time-to-contact, acceleration, and distance that we call the τ-*constraint* and Φ-*constraint*. They allow an active (moving) camera to estimate depth efficiently and accurately while using only a small portion of the image. The constraints are applicable to range sensing, sensor fusion, and visual servoing.

We successfully validate the proposed constraints with two experiments. The first applies both constraints in a trajectory estimation task with a monocular camera and an Inertial Measurement Unit (IMU). Our methods achieve 30-70% less average trajectory error while running 25$\times$ and 6.2$\times$ faster than the popular Visual-Inertial Odometry methods VINS-Mono and ROVIO respectively. The second experiment demonstrates that when the constraints are used for feedback with efference copies the resulting closed loop system's eigenvalues are invariant to scaling of the applied control signal. We believe these results indicate the τ and Φ constraint's potential as the basis of robust and efficient algorithms for a multitude of robotic applications.

2.1 Introduction

Early researchers in computer vision were fascinated by living beings' ability to control their movement in order to gather information about their environment. The process was named "Active Perception" [5–7, 16–18], and numerous mechanisms for utilizing activeness have been developed since. In this work, we focus on the active process of fixation based time-to-contact estimation and using it to measure distance.

Because active vision systems are usually moving in some way, they must accelerate to produce changes to this movement. Despite this, utilizing observer acceleration (measured using an Inertial Measurement Unit or IMU) to facilitate visual computations has not received much attention in the Active Vision literature. To fill this gap, we introduce two mathematical constraints relating (a) time-to-contact (τ) (the ratio of camera velocity to scene distance), (b) the relative size of a planar patch over time, with (c) scene depth and (d) observer acceleration. These constraints make it possible for an active monocular camera to estimate scene depth by accelerating in any direction. We call these constraints the τ-constraint and the Φ-constraint.

We demonstrate the utility of the constraints in a series of experiments involving fixation on a single object (tracking) while a monocular camera accelerates. The method is naturally suited for object-centered representations which have been argued to be useful for a variety of tasks [7, 19–21].

Further, we demonstrate a novel mathematical property of the constraint. When efference copies of the control signal (defined in psychology as an internal copy of the movement producing signal) are used in the place of acceleration in the τ or Φ constraint, the closed loop dynamics become invariant to scaling of the applied control signal. Informally speaking, this means the

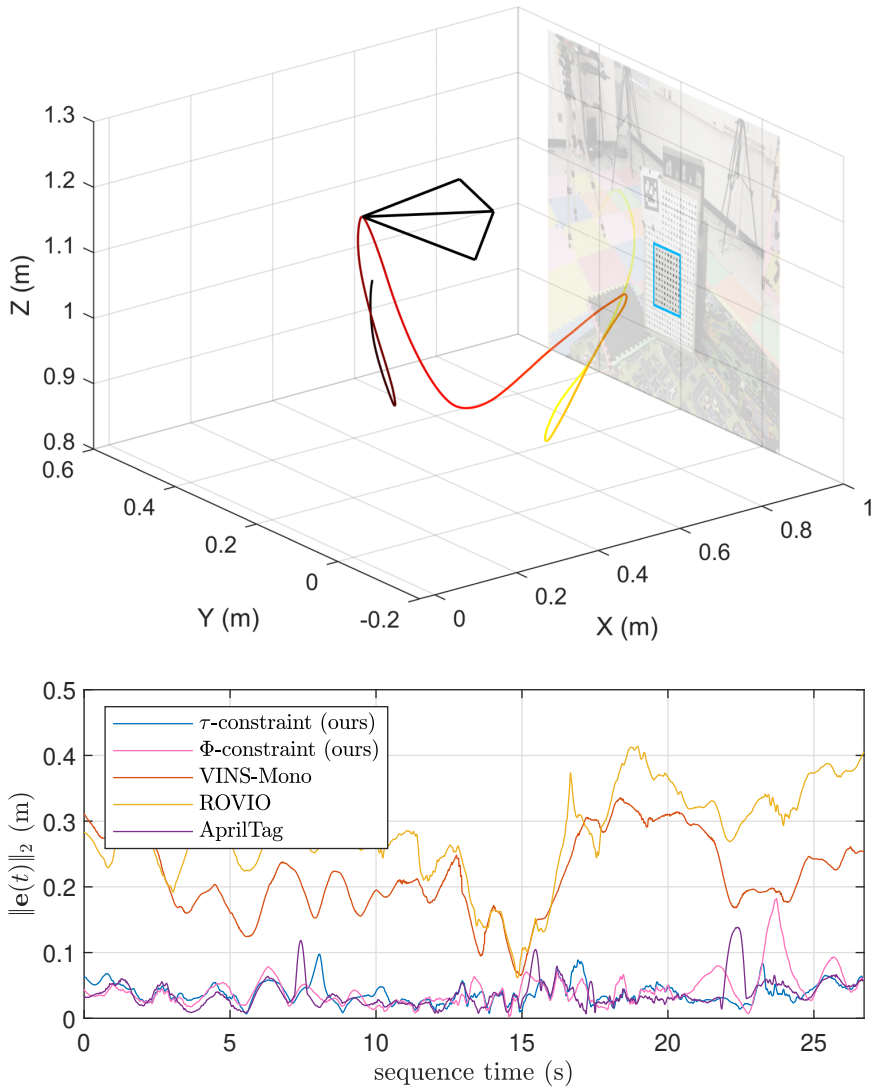

Figure 2.1: Top: Five seconds of Sequence 9's camera trajectory along with the fixated scene patch used to estimate 3D distance using a monocular camera and an IMU. Sequence time is color coded with the hot colormap. Bottom: Instantaneous Euclidean error for Sequence 9 of our methods as well as VINS-Mono, ROVIO, and AprilTag 3. Ground truth is measured by a motion capture system. Supplementary plots for all sequences, and an explanatory video, are available at `https://prg.cs.umd.edu/TTCDist`.

weight of the robot, or the strength of the motors, can change without influencing stability.

The key concept of fusing inertial measurements with camera observations has been extensively studied in the fields of Structure from Motion (SfM) and Simultaneous Localization and Mapping (SLAM). Traditionally, this fusion has been achieved in Visual Inertial Odometry (VIO) frameworks using a series of Bayesian filters or Factor graphs [22, 23].

It is important to stress that our implementation is not a fully-functional VIO implementation, and is only intended to empirically demonstrate the efficacies of the τ and Φ constraints for absolute position estimation and control. Nevertheless, our constraints naturally result in a trajectory estimate. For this reason, we compare our method with the popular state-of-the-art VIO methods such as VINS-Mono [24] and ROVIO [25] as well as the fiducial marker based pose estimation method, AprilTag 3 [26]. We also compare our results with millimeter accurate ground truth from a Vicon motion capture system.

A list of our contributions follows:

- A closed form solution for the 3D position of the camera given time-to-contact or depth-to-scale and acceleration.
- A computationally efficient position estimation method utilizing the τ or Φ constraint based on the closed form.
- Comparisons against the popular VIO methods, VINS-Mono and ROVIO, as well as AprilTag 3, to show the efficacy of the novel constraint in real-world settings.
- A corollary (resulting from the closed form) and experiment showing that our constraints, when used with efference copies in a closed loop, make the system invariant to scaling of the control signal.

2.2 Related Work

A multitude of prior works from both the computer vision and robotics literature deal with time-to-contact and the fusion of camera and IMU measurements to obtain relative camera pose (odometry). However, to the best of our knowledge, none of these works provided closed form solutions for the estimation of distance from time-to-contact and inertial measurements nor for distance from the apparent size of a tracked planar patch and inertial measurements.

Time-to-contact

One of the earliest works to discuss how time-to-contact τ can be used in control tasks came from psychology. [13] provided an analysis of how time-to-contact, τ, obtained from vision, could be used by drivers to control braking a vehicle. The idea was generalized to other perception modalities into a "General Tau Theory" in [27].

Because of its intuitive formulation, τ has also been the subject of many studies in robotics. [28] showed how to land a spacecraft using event cameras by computing τ from the divergence of optical flow. [29] fuses information from a depth camera and τ to compute "time-to-impact" which can in-turn be used to dodge dynamic obstacles. In robotics, most methods that perform optical flow based control use initial height estimates either implicitly or explicitly, as remarked in [30]. To this end, [30] proposed to fuse control effort and time-to-contact with an extended Kalman filter which estimated depth. Another strategy exploits the instability that manifests at certain heights when performing direct τ control with fixed gain feedback to estimate depth [31]. In the context of self-driving cars, BinaryTTC [32] proposed a network to predict per-pixel τ. Recently, EV-Catcher caught fast-moving objects using a network to predict time-to-contact and

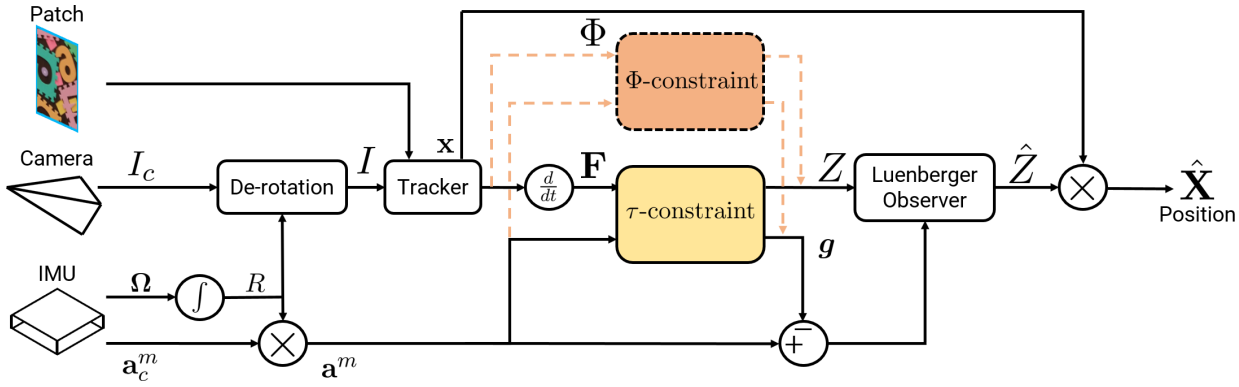

Figure 2.2: System overview of our method to estimate camera position using the τ-constraint. The Φ-constraint uses an identical process except that the derivative of Φ is not taken. Only one constraint is used at a time as indicated by the dashed arrows and outlines.

the current position of a moving object [33]. Finally, it is important to note that τ can be efficiently computed when the scene under consideration is planar [34].

Visual Inertial Odometry (VIO)

One of the earliest works for real-time VIO fused sparse feature tracks with IMU measurements using a Multi-State Constraint Kalman Filter [35]. This pipeline was made more robust by ROVIO [25] which fused photometric consistency of patches with IMU measurements in an Iterative Extended Kalman Filter. ROVIO relies on tracking several planar (affine warpable) patches distributed across the field of view and uses approximations to implement the iterated EKF. To this end, VINS-Mono [24] introduced a point-based nonlinear sliding window estimator with an initialization-free formulation where the points are assumed to be distributed across the field of view. These methods are supported by a proof that tracked points and inertial measurements allow pose to be recovered in closed form [36, 37]. In contrast, our constraints admit closed form solutions for distance to a single patch of unknown size.

Recent deep learning based VIO methods have achieved better accuracy than classical ap-

proaches. VINet [38] presented the first supervised method to estimate VIO using a CNN + LSTM architecture. This was later improved by DeepVIO [39] using supervision from a stereo camera. The limiting factor of these approaches is a lack of speed and generalization across various compute platforms [40]. As will be shown, the τ-constraint can be formulated as a loss, which may be interesting for future work in deep VIO. We will present a derivation of the proposed constraints next.

2.3 Derivation of the τ and Φ constraint

We develop a system to use the Φ and τ constraints with a calibrated monocular camera and an aligned 6-DoF IMU that measures acceleration and angular velocity. Referring to 2.2, we compose the incoming frames with a rotational warp function from the IMU. Then we "fixate on" (track) a planar patch in the rotation-compensated images using an affine homography. A history of the parameters of the affine homography are then used to estimate distance with our proposed Φ and τ constraints. Finally, we filter the predictions using a Luenberger observer (explained in 2.4) to obtain the final trajectory estimates.

In the following derivation, we first define frequency of contact, $\mathbf{F}$, and explain its relation to Φ. Then we show how $\mathbf{F}$ and Φ can be measured from an affine homography. Finally, we related $\mathbf{F}$ and Φ to acceleration and depth which results in the τ and Φ constraints respectively.

2.3.1 Definition of τ, frequency-of-contact $\mathbf{F}$, and Φ

Time-to-contact or τ is defined as $Z/\dot{Z}$ (ratio of depth/distance to velocity). Below, we generalize τ to all three dimensions and define frequency of contact as

$$\mathbf{F} \coloneqq \frac{\dot{\mathbf{X}}}{Z}. \tag{2.1}$$

$\mathbf{X} = [X, Y, Z]^T$ is the position of a point as in 2.3.

Frequency of contact is the number of times per second a constant speed point will reach an axis. Unlike time-to-contact, frequency-of-contact is only ill-defined when $Z = 0$, which just means the object cannot be seen. The third component of $\mathbf{F}$, F_Z, is equal to $1/\tau$.

Frequency-of-contact is directly related to Φ (depth and translation to scale). To show this, consider that $\mathbf{F}$ defines the linear time varying system $\dot{\mathbf{X}} = \mathbf{F}Z$ which has the solution

$$\mathbf{X}(t) = \underbrace{\begin{bmatrix} 1 & 0 & \int_0^t F_X(\lambda)\Phi_{F_Z}(\lambda)d\lambda \\ 0 & 1 & \int_0^t F_Y(\lambda)\Phi_{F_Z}(\lambda)d\lambda \\ 0 & 0 & \Phi_{F_Z}(t) \end{bmatrix}}_{\Phi(t) \coloneqq} \mathbf{X}_0, \tag{2.2}$$

where Φ_{F_Z} is the solution to the ODE $\dot{Z} = F_Z Z$ when $Z_0 = 1$, i.e. $\Phi_{F_Z}(t) = \exp\left(\int_0^t F_Z(\lambda)d\lambda\right)$ [41].

Next, we use a tracked planar patch to estimate $\mathbf{F}$ and Φ.

2.3.2 Estimating τ and Φ from an Affine Homography

As illustrated in 2.3, a point in a scene $\mathbf{X}$ is projected to a point on the image $\mathbf{x}$ according to the pinhole model $\mathbf{x} = \mathbf{X}/Z$. Without loss of generality, we assume that the camera intrinsic matrix K is identity. If we are tracking the points on a plane that is parallel to the imaging plane and the camera translates without rotating, then an affine homography relates all points on the

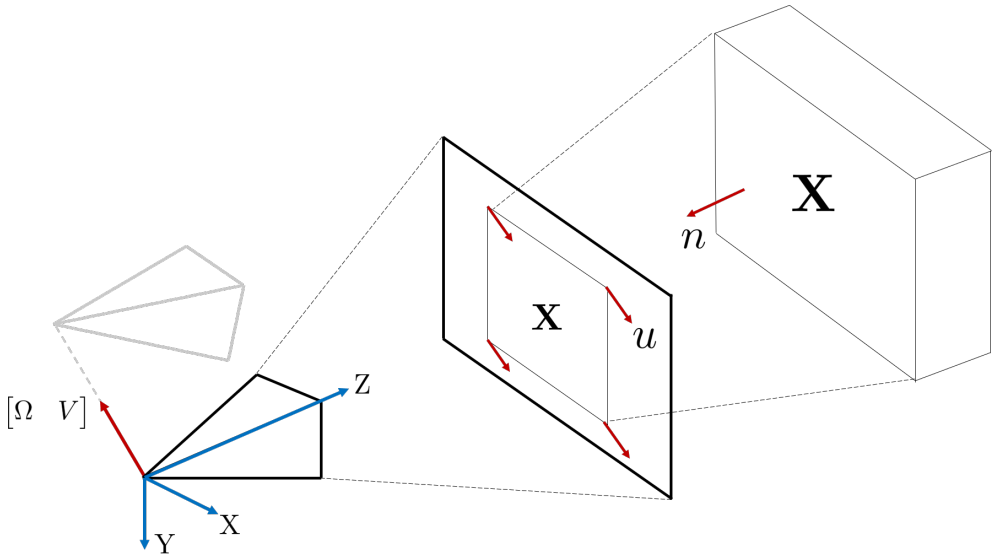

Figure 2.3: By the pinhole model, a scene point $\mathbf{X}$ is projected to image point $\mathbf{x}$. The camera "perceives" movement of $\mathbf{x}$ as optical flow $\mathbf{u}$.

planar patch in the current frame to their positions in the first frame and is given by

$$\mathbf{x}(t) = \underbrace{\begin{bmatrix} Z_0/Z & 0 & (X - X_0)/Z \\ 0 & Z_0/Z & (Y - Y_0)/Z \\ 0 & 0 & 1 \end{bmatrix}}_{A:=} \mathbf{x}(0). \tag{2.3}$$

The definition of Φ in 2.2 allows us to write $\mathbf{X} - \mathbf{X}_0 = (\Phi - I)\mathbf{X}_0$ which reveals that the components of $\Phi - I$ are determined by the elements of A. In our implementation, we estimate the affine homography A using the inverse Lucas-Kanade method. As studied in [42], the patch must have sufficient texture to ensure convergence. We also compose the affine homography with the rotation from inertial measurements (as shown in 2.2).

To obtain $\mathbf{F}$ on the tracked patch from the affine homography we consider the optical flow $\mathbf{u}_\mathbf{x}$ as a function of $\mathbf{x}$

$$\mathbf{u}_{\mathbf{x}} = \frac{d\mathbf{x}(t)}{dt} = \dot{A}A^{-1}\mathbf{x} = -\underbrace{\begin{bmatrix} \dot{Z}/Z & 0 & \dot{X}/Z \\ 0 & \dot{Z}/Z & \dot{Y}/Z \\ 0 & 0 & 0 \end{bmatrix}}_{B\coloneqq} \mathbf{x}. \tag{2.4}$$

Thus, B's terms are equal to frequency-of-contact $\mathbf{F}$. In practice, we fit an affine homography to a slightly slanted plane. In that case, it is more appropriate to consider the affine flow parameters in $B = [b_{i,j}]$ as corresponding to the linear terms of optical flow due to a planar surface. Then, if $\eta \coloneqq ((b_{2,1}b_{1,3}/b_{2,3}) - b_{1,1})$, the frequency-of-contact is

$$\frac{\dot{\mathbf{X}}}{Z} = \begin{bmatrix} b_{2,1}b_{1,3}/b_{2,3} & b_{1,2} & b_{1,3} \\ b_{2,1} & b_{1,2}b_{2,3}/b_{1,3} & b_{2,3} \\ \eta(b_{2,1}/b_{2,3}) & \eta(b_{1,2}/b_{1,3}) & \eta \end{bmatrix} \mathbf{x}. \tag{2.5}$$

Next, our constraints are derived from the above relations.

2.3.3 The τ-constraint and the Φ-constraint

Now, we will relate $\mathbf{F}$ and Φ to depth and acceleration which results in the τ and Φ constraints respectively. The constraints relate the initial conditions of the linear time varying system defined by time-to-contact and the linear time invariant system defined by acceleration.

By the fundamental theorem of calculus, $\mathbf{X}$ is given by

$$\mathbf{X}(t) - \mathbf{X}_0 = t\dot{\mathbf{X}}_0 + \underbrace{\int_0^t \left(\int_0^\lambda \ddot{\mathbf{X}}(\lambda_2) d\lambda_2 \right) d\lambda}_{\mathcal{J}\{\ddot{\mathbf{X}}\}(t)\coloneqq}. \tag{2.6}$$

Where $\mathcal{J}\{\mathbf{f}\}(t) : L_p^n \times \mathbb{R} \to \mathbb{R}^n$ is an operator returning the double integral of a vector of functions [43].

Since $\mathbf{X} - \mathbf{X}_0 = (\Phi - I)\mathbf{X}_0$ we can make a substitution which results in the Φ constraint.

Theorem 2.3.1 (Φ-constraint)**.** *If Φ and $\ddot{\mathbf{X}}$ are known $\forall t$ in a closed time interval and $Z \geq \epsilon > 0, \forall t$, then the following linear constraint between initial depth Z_0, Φ, and acceleration $\ddot{\mathbf{X}}$ holds for each point on the planar scene patch*

$$(\Phi(t) - I) \begin{bmatrix} 0 \\ 0 \\ Z_0 \end{bmatrix} - t\dot{\mathbf{X}}_0 = \mathcal{J}\{\ddot{\mathbf{X}}\}(t). \qquad (\Phi\text{-constraint})$$

Using the Φ-constraint for position estimation requires determining four unknowns, Z_0 and $\dot{\mathbf{X}}_0$. Using the fact that $(\Phi(t) - I) \propto t$ if and only if $\ddot{\mathbf{X}} = 0, \forall t$ makes it possible to show that when $\Phi(t)$ and $\mathcal{J}\{\ddot{\mathbf{X}}\}(t)$ are known over a time interval, then the linear system defined by the Φ-constraint determines Z_0 and $\dot{X}_0$ if and only if $\exists\, t$ between the two times s.t. $\ddot{\mathbf{X}} \neq 0$. A proof is in the supplementary material.

Since $\dot{\mathbf{X}}_0 = \mathbf{F}(0)Z_0$, the τ-constraint follows directly.

Theorem 2.3.2 (τ-constraint)**.** *If $\mathbf{F}$ and $\ddot{\mathbf{X}}$ are known $\forall t$ in a closed interval and $Z \geq \epsilon > 0, \forall t$, then the following linear constraint between depth Z_0, frequency-of-contact $\mathbf{F}$, and acceleration*

$\ddot{\mathbf{X}}$ *holds for each point on the planar scene patch*

$$\underbrace{\left(\Phi(t) - I - t\begin{bmatrix} 0 & 0 & \mathbf{F}(0) \end{bmatrix}\right)}_{E(t):=} \begin{bmatrix} 0 \\ 0 \\ Z_0 \end{bmatrix} = \mathcal{J}\{\ddot{\mathbf{X}}\}(t). \qquad (\tau\text{-constraint})$$

$E(t)$ is the ratio between positional change due to acceleration and the depth Z_0. Intuitively, it is the *"action's effect"*.

It is proven in the supplementary material that determining Z_0 is well posed if and only if $\ddot{\mathbf{X}}$ is non-zero at some time.

The Φ-constraint and τ-constraint are closely related. The Φ-constraint considers the position and area of an object in the image, and the τ-constraint considers its velocity and rate of change of size. Since $\mathbf{F}$ determines Φ by integration, it is reasonable to use either the τ or Φ constraint when $\mathbf{F}$ is available. However, using the τ-constraint when only Φ is available is difficult because Φ must be numerically differentiated to get $\mathbf{F}$, which can introduce significant noise.

Next, we discuss estimating depth using our constraints.

2.4 Fusing Inertial Measurements with τ and Φ

To tightly couple IMU measurements and frequency-of-contact from a camera in a sliding window manner, we set up a least squares problem over the constraints. The solution is the depth of a point in the scene and the gravity direction.

As illustrated in 2.2, we estimate a rotation matrix, R, using the gyroscope. Using this

matrix allows all measurements to be rotated into the orientation of the initial camera frame, and thus over a short time period, the method can be considered rotation invariant. Thus, we use R to rotate the measured acceleration $\mathbf{a}_c^m(t)$ back to a fixed frame, $\mathbf{a}^m(t) = R(t)\mathbf{a}_c^m(t)$. The IMU measures the resultant of gravitational acceleration and linear acceleration. Thus, $\mathbf{a}^m(t) = -\ddot{X} + \boldsymbol{g}$.

2.4.1 Efficient Computation of Depth (Z Distance)

Let us suppose a history of $\mathbf{a}^m(t)$ and $\mathbf{F}(t)$ measurements are available over a time interval $[0, T]$. Now, without loss of generality, we consider the problem along only the Z axis. The problem for the τ-constraint can be written as

$$\underset{Z_0, g_Z}{\operatorname{argmin}} \int_0^T \left(E_Z Z_0 + \mathcal{J}\{a_Z^m + g_Z\}\right)^2 dt. \tag{2.7}$$

The problem for the Φ-constraint is given by

$$\underset{Z_0, \dot{Z}_0, g_Z}{\operatorname{argmin}} \int_0^T \Big(\left(\Phi_{F_Z} - 1\right) Z_0 - r\dot{Z}_0 + \mathcal{J}\{a_Z^m + g_Z\}\Big)^2 dt, \tag{2.8}$$

where $r(t) \coloneqq t$ is the ramp function.

In both cases, the problem is the same for the X or Y axis up to a change of subscripts and the initial velocity.

It is proven in the supplementary material that 2.7 and 2.8 are well-posed if and only if the measured acceleration, a_z^m, is not constant for the entire time interval.

Thus, given motion along an axis, depth and gravitational acceleration can be recovered efficiently by solving a linear least squares problem based on either constraint. The estimates

have noise and so we discuss filtering them next.

2.4.2 Filtering of Depth

Now, we extend the above derivation for trajectory estimation. Since 2.7 estimates g_z and Z, we can set up a Luenberger observer [41] to filter the trajectory estimate. Let $\hat{Z}$ and $\hat{\dot{Z}}$ be the estimated quantities, then

$$\begin{bmatrix} \dot{\hat{Z}} \\ \ddot{\hat{Z}} \end{bmatrix} = \begin{bmatrix} \hat{\dot{Z}} \\ a_z^m - g_z \end{bmatrix} + L \begin{bmatrix} Z - \hat{Z} \\ \dot{Z} - \hat{\dot{Z}} \end{bmatrix}. \tag{2.9}$$

When multiple estimates for Z are available from motion along multiple axes, the results are averaged. When $\hat{Z}$ and g_z are not available, due to lack of acceleration, then dead reckoning is used by applying $\mathbf{F}$ or Φ to the latest $\hat{Z}$.

Finally, we recover the three dimensional trajectory by multiplying $\hat{Z}$ with the current image location of the center of the planar patch: $\begin{bmatrix} \hat{X} & \hat{Y} & \hat{Z} \end{bmatrix}^T := \mathbf{x}\hat{Z}$. Next, we explain how to substitute control effort for acceleration.

2.5 Closed Loop control using Efference Copies

Since the control effort in robotics often corresponds to acceleration up to some scale, it is interesting to consider what happens when control effort, $\mathbf{u}$, is substituted for $\ddot{\mathbf{X}}$ when solving 2.7. Control effort and optical flow are often referred to as $\mathbf{u}$. So we use $\mathbf{u}_\mathbf{x}$ for the flow at pixel $\mathbf{x}$.

This substitution leads to an interesting property.

Corollary 2.5.1. *If* $\ddot{\mathbf{X}} = b\mathbf{u}, b \neq 0$*, where* $\mathbf{u}$ *is a linear control determined by* $\mathbf{u} = K\hat{\mathbf{X}}$*, then if* $\mathbf{u}$ *is substituted for* $\mathbf{a}^m$ *in 2.7, the dynamics become invariant to* b*.*

Proof. By definition $\ddot{\mathbf{X}}/b = \mathbf{u}$, and the solution to 2.7 is linear in $\mathbf{a}^m$. Thus, using $\mathbf{u}$ instead of $\mathbf{a}^m$ results in a state estimate scaled by the inverse of b, i.e. $\hat{\mathbf{X}} = \mathbf{X}/b$.

Further, since $\mathbf{u} \coloneqq K\hat{\mathbf{X}}$, we can see that $\ddot{\mathbf{X}} = bK(\mathbf{X}/b) = K\mathbf{X}$. Thus, the dynamics are invariant to b.

□

Practically speaking, this allows changing the strength of a motor, or the weight of a robot, without drastically changing the stability properties of the system. Typically such a change would require re-tuning the control gains K.

2.6 Experiments

We designed the experiments to evaluate the τ and Φ constraints as well as the invariance property to determine if the constraints are a promising method for future research and applications.

2.6.1 Metric Trajectory Estimation

To test the constraints' ability for trajectory estimation, we created ten sequences with five distinct scenes in an indoor setting. Each scene contains a planar object to fixate on as shown in 2.1. For each scene, two recordings were made, one with an AprilTag and one without. The trajectories feature acceleration often over 2 $\mathrm{m/s^2}$.

The sequences were recorded with an Intel® RealSense™ D435i camera using the built-in IMU and the left grayscale camera [44]. We used the grayscale camera because the D435i's IMU is hardware time-stamped to the grayscale imager. The D435i captures images at 90 frames per second at 848 $\times$ 480 px. resolution. The IMU records gyroscope measurements at 400 Hz and the acceleration at 250 Hz.

We tracked the fixated planar patch using gyroscope measurements for rotation stabilization and by continually fitting an affine homography using the ubiquitous patch tracking method from [42]. The frequency-of-contact $\mathbf{F}$ was then recovered using 2.5. The affine tracker was initialized by tracking a 100$\times$100 pixel patch sub-sampled to 4000 pixels. While the patch size changes dramatically during fixation, only 4000 pixels are drawn from each frame.

Then, the τ-constraint was fused with the IMU measurements using a 2 second signal history and a 100 Hz sampling rate using linear interpolation. If the average power of the bias corrected acceleration along an axis was below 2 $\mathrm{m/s^2}$ the resulting depth from the τ-constraint was not used. If no observations of depth were available the depth estimate was forward propagated using dead reckoning with $\mathbf{F}$ or Φ. The Luenberger's gain was set to $L = \mathrm{diag}(2, 20)$ for all the sequences. *Also, it is important to note, that these parameters were not tuned using the sequences considered in this work.*

Our implementation is written in Python 3.8 using standard scientific Python libraries. Numba is used to accelerate critical sections [45]. One thread on an Intel® Core™ i7-6820HQ laptop processor was used to perform computations.

The open-source VINS-Mono and ROVIO implementations were used for comparison [24, 25, 46]. VINS-Mono was configured to output poses at 90 Hz (the camera frame rate), however, it produced poses only at 80 Hz. ROVIO was configured to produce poses at 90 Hz.

As another source of comparison, we used the AprilTag 3 [26] library to detect 36h11 tags. The corners were used to solve the Perspective-n-Point problem to recover a tag's location in the current frame. The gyroscope measurements were used to rotate each AprilTag pose measurement back to the fixed orientation used by the constraints.

The ground truth trajectory was measured at 200 Hz using a Vicon motion capture system with 8 Vantage V8 cameras. We align all trajectories to Vicon ground truth and compute the Average Trajectory Error (ATE) as described in [47].

$$\text{ATE}(\mathbf{X}, \mathbf{X}^v) = \left(\frac{1}{N} \sum_{n=0}^{N-1} \|\mathbf{X}_n - \mathbf{X}_n^v\|_2^2 \right)^{1/2}. \tag{2.10}$$

Here $\mathbf{X}_n^v$ is themotion capture systems n'th position estimate and $\mathbf{X}_n$ is the n'th estimated position.

2.6.2 Closed Loop Stability Invariance Property

To test the invariance property, we implement a closed loop controller on a DJI® RoboMaster™ robot pictured in Fig. 2.4. The robot's goal is to reach a fixed distance from a pre-determined visual target, where distance is in meters for trials using measured acceleration and units of effort when using efference copies. By adjusting a static gain b applied to the control signal we can test if using efference copies in the Φ-constraint prevents the control system from exhibiting unstable behavior when b is increased dramatically. Four trials were run. In two of these acceleration from the onboard IMU was fed to the Φ-constraint whose estimates were then used for closed loop control. In the other two trials, efference copies were fed to the Φ-constraint, meaning that $\mathbf{u}$ was used instead of the measured acceleration. For each group of two experiments, one was run with

Seq.	1	2	3	4	5
Duration (s)	15.06	26.15	32.28	36.23	16.41
Length (m)	15.73	29.65	22.21	34.75	15.63
Method	ATE (cm)				
AprilTag 3	**2.80**	-	**2.67**	-	**3.76**
VINS-Mono	5.41	8.80	14.21	15.37	-
ROVIO	7.77	9.89	11.88	33.23	29.96
Φ-constraint (ours)	**3.77**	**5.79**	**7.60**	**7.32**	**7.40**
τ-constraint (ours)	8.07	6.91	12.33	10.21	16.82
Seq.	6	7	8	9	10
Duration (s)	16.27	8.02	32.15	26.73	40.08
Length (m)	15.78	7.30	26.75	21.39	35.37
Method	ATE (cm)				
AprilTag 3	-	**0.65**	-	**2.48**	-
VINS-Mono	6.10	1.15	18.45	13.07	4.34
ROVIO	**2.84**	**0.69**	3.93	16.62	3.79
Φ-constraint (ours)	5.71	1.42	**3.28**	2.86	**2.42**
τ-constraint (ours)	7.21	10.70	4.32	**2.38**	3.24

Table 2.1: Sequence duration (seconds), path length (meters), and each method's accuracy in centimeters of Average Trajectory Error (ATE). Since AprilTag 3 is always the best when it is available we also bold the second best result in such sequences.

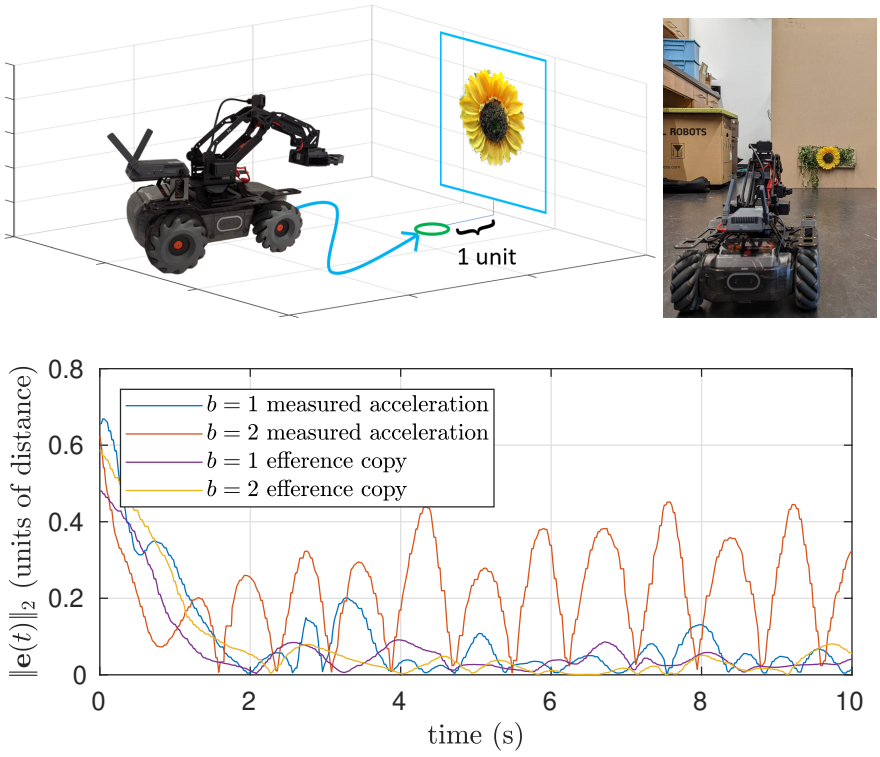

Figure 2.4: Top left: The invariance property is tested by using the constraint to navigate to a unit distance from a visual target. Top right: The experimental setup. Bottom: The normed error between the robot's position and the target position. When efference copies are used, the system is stable despite changes to the unknown gain b. When using measured acceleration, the system destabilizes when b is set to 2. Distance is in meters for trials using measured acceleration and units of effort when using efference copies.

the actuator gain set to 1 (which was the value the K matrix was tuned for). In the second run, the actuator gain was set to 2, which doubled all control signals without the control algorithm's knowledge. The control gains were chosen as $K = \text{diag}(2, 2)$.

2.7 Results And Discussion

2.7.1 Metric Trajectory Estimation

The ATE results across all the 10 sequences are given in 2.1 along with the path length and duration of the sequences. The proposed Φ-constraint achieves lower ATE than VINS-Mono

and ROVIO in all but two sequences. In those two sequences, it remains competitive. The τ-constraint achieves a lower ATE than VINS-Mono in six out of nine comparable sequences and a lower ATE than ROVIO in 5 out of 10 comparable sequences. ATE averaged over all sequences was 5.4 cm for the Φ-constraint, 8.5 cm for the τ-onstraint, 16.9 cm for ROVIO, and 2.8 cm for AprilTag 3. The average ATE for VINS-Mono, averaged over all sequences except sequence 5, was 12.2 cm. The VINS-Mono result is omitted for sequence 5 because its estimate diverged.

It is no surprise that the AprilTag 3 method routinely achieved the best ATE. This is because the AprilTag system uses the known size of the visual fiducial to estimate depth. Regardless, the τ and Φ-constraints perform comparably to the AprilTag 3 method in some sequences. This is particularly noticeable in Sequence 9. In 2.1 the instantaneous l_2 error of each method in Sequence 9 is plotted for comparison.

While the ATE errors are promising, they do not indicate that our method is better than existing VIO methods. Such a claim would require developing a full VIO stack around the τ or Φ-constraint and comparisons on existing datasets.

Our Python implementation achieved 6.5 $\times$ realtime or 588 frames per second (fps). VINS-Mono's C++ implementation ran at 0.26 $\times$ realtime or 23.6 fps. ROVIO's C++ implementation ran at 1.05 $\times$ realtime or 94.5 fps.

2.7.2 Closed Loop Stability Invariance Property

As shown in Fig. 2.4, all achieved trajectories are similar, and approach the target, except for the case where the actuator gain was doubled and measured linear acceleration was used in the Φ-constraint. This was expected. Indeed, in this case, the poles and zeros of the closed

loop system are dramatically shifted because the control gain matrix is effectively doubled. As a result, the robot began to oscillate around its stopping point as is typical of a "poorly tuned" controller. However, the control scheme using efference copies had no such limitation, as predicted by Corollary 2.5.1.

2.8 Conclusion and Future Work

In our work, we developed two novel constraints called the τ and Φ-constraint which allow a moving camera to estimate depth using a small part of the image. Applying these constraints to trajectory estimation achieved better results while being orders of magnitude faster than some state-of-the-art VIO approaches. Further, we presented a method to perform closed-loop control with the constraints while using efference copies which is invariant to scaling of the control signal. We will talk about some future directions next.

Both constraints require that there is acceleration in order to measure distance. In practice, we found accelerations with approximately 2 m/s^2 of power were necessary to get good measurements. However, when using efference copies only a small nominal acceleration was required for reasonable performance. Further theoretical analysis would be useful for gaining more insight into this behaviour.

VIO methods commonly estimate IMU biases and so it would be interesting to add such terms to our constraints. Similarly, it is of interest to extend 2.5.1 to account for a transfer function relating control effort and acceleration.

For our method to be deployable as a full VIO system it would need to be extended to fixate on multiple patches and actively switch between them. Based on the significant speed up, and

competitive accuracy presented in our preliminary results, we believe that further development of the τ and Φ constraint, in theory and practice, is a promising direction for VIO, VI-SLAM, active perception, and robotics.

2.9 Proofs

2.9.1 Completion of Proof of Theorem 2.3.1

For brevity, the manuscript left out the proof that the Φ-constraint allows estimating initial depth Z_0 and initial velocity $\dot{\mathbf{X}}_0$ given measurements in an interval $t \in [0, T]$ if and only if $\exists\ t$ between the two times s.t. $\ddot{\mathbf{X}} \neq 0$. In what follows we assume $\mathbf{X}(t)$ is sufficiently differentiable.

Proof. Recall the Φ-constraint is

$$(\Phi(t) - I) \begin{bmatrix} 0 \\ 0 \\ Z_0 \end{bmatrix} - t\dot{\mathbf{X}}_0 = \mathcal{J}\{\ddot{\mathbf{X}}\}(t). \qquad (\Phi\text{-constraint})$$

These three constraints are linear in Z_0 and $\mathbf{X}_0$. Thus, solving for these quantities using measurements over an interval $t \in [0, T]$ is not well posed if and only if $\Phi(t) - I$ and $-t$ are linearly dependent over the interval.

Consider that

$$\begin{aligned}
&\mathbf{X}(t) = \Phi(t)\mathbf{X}_0 \\
\iff &\mathbf{X}(t) - \mathbf{X}_0 = (\Phi(t) - I)\mathbf{X}_0 \\
\iff &\mathbf{X}(t) - \mathbf{X}_0 = (\Phi(t) - I) \begin{bmatrix} 0 \\ 0 \\ Z_0 \end{bmatrix} \\
\iff &\frac{\mathbf{X}(t) - \mathbf{X}_0}{Z_0} = (\Phi(t) - I) \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.
\end{aligned} \tag{2.11}$$

Therefore it is equivalent to determine when $(\mathbf{X}(t)-\mathbf{X}_0)/Z_0$ and $-t$ are linearly dependent. Assuming the two functions are equal up to a scalar $\alpha \in \mathbb{R}$ reveals

$$\begin{aligned}
&\frac{\mathbf{X}(t) - \mathbf{X}_0}{Z_0} = \alpha(-t) \\
\iff &\mathbf{X}(t) = \mathbf{X}_0 + \alpha(-t)Z_0 \\
\iff &\ddot{\mathbf{X}}(t) = 0.
\end{aligned} \tag{2.12}$$

Thus, the functions are linearly independent, and so Z_0 and $\mathbf{X}_0$ can be determined, if and only if $\exists\, t \in [0, T]$ s.t. $\ddot{\mathbf{X}}(t) \neq 0$. □

2.9.2 Completion of Proof of Theorem 2.3.2

For brevity, the manuscript left out the proof that the τ-constraint allows estimating initial depth Z_0 given measurements in an interval $t \in [0, T]$ and if and only if $\exists\ t$ between the two times s.t. $\ddot{\mathbf{X}} \neq 0$.

Proof. Recall the τ-constraint is

$$\underbrace{\left(\Phi(t) - I - t\begin{bmatrix} 0 & 0 & \mathbf{F}(0) \end{bmatrix}\right)}_{E(t) :=} \begin{bmatrix} 0 \\ 0 \\ Z_0 \end{bmatrix} = \mathcal{J}\{\ddot{\mathbf{X}}\}(t). \qquad (\tau\text{-constraint})$$

The constraint is linear in Z_0 thus solving for Z_0 quantities using measurements over an interval $t \in [0, T]$ is not well posed if and only if $E(t) = 0\ \forall\ t \in [0, T]$. Substituting the previous expression for $\Phi(t) - I$ and the definition of $F(0) = \dot{\mathbf{X}}_0 / Z_0$ reveals

$$\begin{aligned} & \Phi(t) - I - t\begin{bmatrix} 0 & 0 & \mathbf{F}(0) \end{bmatrix} = 0 \\ \iff & \frac{\mathbf{X}(t) - \mathbf{X}_0}{Z_0} - t\frac{\dot{\mathbf{X}}_0}{Z_0} = 0 \\ \iff & \mathbf{X}(t) = \mathbf{X}_0 + t\dot{\mathbf{X}}_0 \\ \iff & \ddot{\mathbf{X}}(t) = 0 \end{aligned} \qquad (2.13)$$

Therefore, the problem is well-posed (meaning Z_0 can be determined) if and only if $\exists\ t \in [0, T]$ s.t. $\ddot{\mathbf{X}}(t) \neq 0$. □

2.9.3 Proof that (2.7) and (2.8) are well posed

Equations (7) and (8) setup a linear least squares problem to determine Z_0, $\dot{Z}_0$ and g_Z from $\Phi(t) - I$ and a_Z^m. It is stated the problem is the same for the X or Y axis up to a change of subscripts and the initial velocity and claimed without proof that the problems are well-posed if and only if, a^m is not constant for the entire time interval. The proof follows.

Proof. For Eq. (7), the linear relation being optimized (based on the Φ-constraint) and considered along each axis at once is

$$\begin{aligned} &(\Phi(t) - 1) \begin{bmatrix} 0 \\ 0 \\ Z_0 \end{bmatrix} - t\dot{\mathbf{X}}_0 + \mathcal{J}\{\mathbf{a}^m + \mathbf{g}\}(t) = 0 \\ \iff &\frac{\mathbf{X}(t) - \mathbf{X}_0}{Z_0} Z_0 - t\dot{\mathbf{X}}_0 + \mathbf{g}\frac{t^2}{2} + \mathcal{J}\{\mathbf{a}^m\}(t) = 0. \end{aligned} \tag{2.14}$$

Similarly to Theorem 3.1 and 3.2, the problem is not well posed if and only if $(\mathbf{X}(t) - \mathbf{X}_0)/Z_0$, $-t$, and $t^2/2$ are linearly dependent $\forall\ t \in [0, T]$. Assuming $\exists\ \alpha, \beta \in \mathbb{R}$ s.t. a weighted linear combination equals zero reveals

$$\begin{aligned} &\frac{\mathbf{X}(t) - \mathbf{X}_0}{Z_0} - \alpha t + \beta\frac{t^2}{2} = 0 \\ \iff &\mathbf{X}(t) = \mathbf{X}_0 + \alpha Z_0 t - \beta Z_0 \frac{t^2}{2} = 0 \\ \iff &\frac{d}{dt}\ddot{\mathbf{X}}(t) = 0. \end{aligned} \tag{2.15}$$

The proof and result for Eq. (9) (based on the τ-constraint) is identical with the exception that $\alpha Z_0 t$ in the second to last line becomes $\alpha \dot{\mathbf{X}}_0 t$.

Thus, the problems in Eq. (8) and (9) are well-posed if and only if the derivative of acceleration, jerk, is non-zero at some time during the considered time interval. □

Chapter 3: Embodied Visuomotor Representation

Imagine sitting at your desk, looking at objects on it. You do not know their exact distances from your eye in meters, but you can immediately reach out and touch them. Instead of an externally defined unit, your sense of distance is tied to your action's embodiment. In contrast, conventional robotics relies on precise calibration to external units, with which vision and control processes communicate. We introduce *Embodied Visuomotor Representation*, a methodology for inferring distance in a unit implied by action. With it a robot without knowledge of its size, environmental scale, or strength can quickly learn to touch and clear obstacles within seconds of operation. Likewise, in simulation, an agent without knowledge of its mass or strength can successfully jump across a gap of unknown size after a few test oscillations. These behaviors mirror natural strategies observed in bees and gerbils, which also lack calibration in an external unit.

3.1 Introduction

The predominant autonomy frameworks in robotics rely on calibrated 3D sensors, predefined models of the robot's physical form, and structured representations of environmental interactions. These representations allow vision and low-level control to be abstracted as separate processes that rely on an external scale, such as the meter, to coordinate. For instance, it is

common to use vision to construct a metric map scaled to the meter. Subsequently, a planning algorithm uses this geometric representation to generate a trajectory, also scaled to the meter. Finally, a pre-tuned low-level controller employs feedback to follow the metric trajectory, mapping it to motor signals. This approach is known as the sense-plan-act paradigm, and it originates from Marr's vision framework [4]. Figure 3.1 illustrates a block diagram of this paradigm.

The sense-plan-act architecture allows separate teams of engineers and scientists to create equally separate vision and control algorithms tuned for particular tasks and mechanical configurations. Subsequently, alternative frameworks for vision, such as Active Vision and Animate Vision emerged [5, 19, 48], primarily to address the limitations of the passive role of vision in the sense-plan-act cycle. However, these frameworks consider control (action) at a high level, and so vision and control remain mostly separate fields and continue to be interfaced with external scale. This dependence leads to lengthy design-build-test cycles and, consequently, expensive systems that are available in only a few physical configurations and must be precisely manufactured. Further, it creates problems that are unobserved in biological systems. For example, it is well-known that Advanced Driver-Assistance Systems (ADAS) in modern cars, such as lane-keeping assist and collision warning, require millimeter accurate, per vehicle calibrations. Another example is the 1999 NASA Mars Climate Orbiter, whose software confused English and Metric units and crashed [49]. Finally, several billion-dollar 3D Camera and LIDAR industries aim to produce accurate 3D measurements calibrated to an external scale, and significant effort is dedicated to ensuring those calibrations do not degrade over time.

This situation is striking because biological systems do not necessarily understand distance in an external scale like the meter. Further, we know from psychology that human and other mammalian perceptual systems do not provide metric depth and shape [50–54]. Instead, somehow,

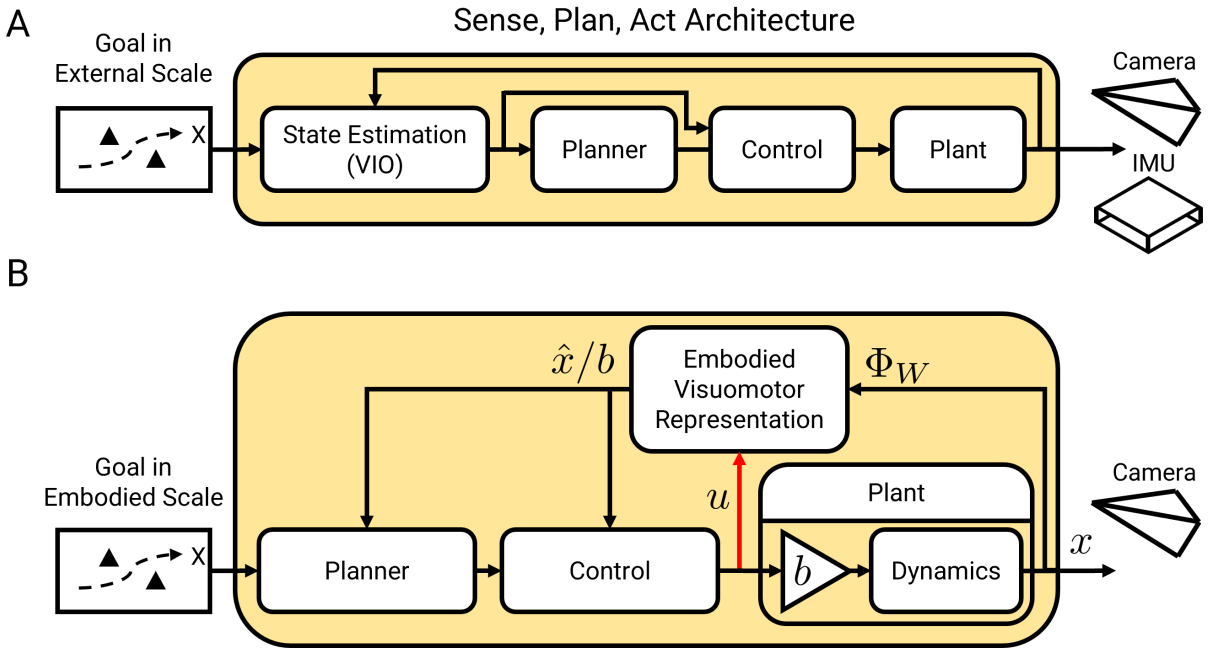

Figure 3.1: **Architectural comparison of sense-plan-act with Embodied Visuomotor Representation.** (A): The classic sense-plan-act architecture used in robotics assuming visual inertial odometry (VIO) is used for state estimation. Stability depends on calibrated sensors, such as an IMU, that provide accurate knowledge of the state in an external scale. (B): An architecture based on Embodied Visuomotor Representation. Compared to sense-plan-act, the embodied approach includes an additional internal feedback connection (red arrow) containing the control signal u. The units of u are implied by the unknown gain b and the dynamics going from control to the state in any scale, including the meter. Embodied Visuomotor Representation leverages position-to-scale, Φ_W, obtained from vision and u to determine a state estimate $\hat{x}/b$ in the embodied scale of u. Notably, the unknown b cancels in the closed-loop system, enabling stable control without calibrated sensors. Direct methods such as Tau Theory, Direct Optical Flow Regulation, and Image Based Visual Servoing also avoid dependence on calibrated sensors by using purely visual cues (e.g., time-to-contact, optical flow, or tracked image features) for feedback. However, with few exceptions, such methods' stability depends on tuning the control law for the expected scene distances and system velocities.

biological systems represent the world using only the signals due to their embodiment. Further, animals such as gerbils, dragonflies, and bumblebees exhibit visuomotor capabilities far in excess of robots while allowing for much broader variation in physical form and handling complex dynamics such as turbulence and muscular response [14, 15, 55]. Similarly, humans can control augmentations to their embodiment, such as cars and airplanes, which also vary in physical form and do not maintain precise calibrations to an external scale.

One explanation is that these systems do not rely on any precise sense of distance. In support, direct approaches such as Tau Theory [13], Direct Optical Flow Regulation [56], and classical Image Based Visual Servoing [57] offer methods to control a visuomotor system using only cues directly available from the image such as time-to-contact, optical flow, and image features. However, systems strictly limited to direct methods cannot reason about distance, a capability essential for behaviors such as the clearing maneuvers of bees [14] and the jumping ability of gerbils [15]. Moreover, direct methods either assume a predefined operating region for which the control system is tuned [57–60] or constrain achievable behaviors to those possible with a restricted class of feedback laws [61, 62]. In contrast, systems with knowledge of distance can overcome these limitations while still using direct (image space) control objectives.

Thus, a key question arises: How can a robot, without a source of external scale, use its vision and motor system to estimate a meaningful sense of distance? We refer to methods that capture this phenomenon as *Embodied Visuomotor Representation* and propose that its realization could drive a paradigm shift that makes robots easier to produce, more affordable, and widely accessible.

Psychologists assume humans have such representations without proposing a precise and general method for obtaining them [63]. Biologists argue that simple animals such as bees can

develop them [14]. Computer scientists suggest that computations drive intelligence, and thus representations, as physical as the body itself [64]. In particular, there is evidence that the visuomotor capabilities of animals can be attributed to well-tuned internal models closely coupled with perceptual representations. In addition to reasoning about distances, an important function of these models is prediction, which allows an animal to anticipate upcoming events, compensate for the significant time delays in their feedback loops, and, when coupled with inverse models, generate feedforward signals for other parts of the body [65].

However, being able to explain a phenomenon is not the same as being able to implement it. To address this, we propose Embodied Visuomotor Representation, a concrete framework that can estimate embodied, action-based distances using an architecture that couples vision and control as a single algorithm. This results in robots that can quickly learn to control their visuomotor systems without any pre-calibration to a unit of distance. In many ways, our method parallels a psychological theory of memory as embodied action due to Glenberg (who presents the idea of embodied grasping of items on a desk). Glenberg argues that the core benefit of embodied representation is that they "do not need to be mapped onto the world to become meaningful because they arise from the world" [63]. Similarly, our Embodied Visuomotor Representations do not need to be pre-initalized with an external scale because they can be estimated by the robot online without sacrificing stability.

Embodied Visuomotor Representation achieves this by exchanging distance on an external scale for the unknown but embodied distance units of the acceleration affected by the motor system. As a consequence, powerful self-tuning, self-calibration, or adaptive properties emerge. At the mathematical level, the approach is similar to the Internal Model Principle, a control theoretic framework that underlies all learning-based control methods, either explicitly or implicitly [66].

Unlike the sense-plan-act approach, which operates sequentially, the Internal Model Principle introduces a critical topological distinction: an internal feedback loop that leverages embodied motor signals to predict sensory feedback representations. This same internal feedback loop, shown in Figure 3.1 enables Embodied Visuomotor Representation to integrate vision and control seamlessly, without relying on external scale, or sacrificing close-loop stability.

In what follows, we develop a general mathematical form for equality constrained Embodied Visuomotor Representation and demonstrate it on a series of experiments where uncalibrated robots gain the ability to touch, clear obstacles, and jump gaps after a few seconds of operation by using natural strategies observed in bees flying through openings and gerbils jumping gaps. For these applications, we develop specific relations between control inputs, acceleration, and observed visual features, such as lines and planes, that can be used by an uncalibrated robot to estimate state in an embodied unit. This state is used by control schemes that are guaranteed to perform a task correctly. The results suggest that Embodied Visuomotor Representation is a paradigm shift that will allow all visuomotor systems, and in particular robots, to automatically learn to control themselves and engage in lifelong adaption without relying on pre-configured 3D sensors, controllers, or 3D models that are calibrated to an external scale.

3.2 Results

First, the mathematical methodology for Embodied Visuomotor Representation will be detailed in the equality constrained case, where the results of vision and control can be set equal to each other. Next, examples of applying Embodied Visuomotor Representation to the control of a double integrator and a multi-input system with actuator dynamics are developed. Finally,

algorithms that use Embodied Visuomotor Representation to allow uncalibrated robots to learn to touch, clear obstacles, and jump gaps are presented. Detailed calculations behind these applications are provided in the Methods section.

3.2.1 Equality Constrained Embodied Visuomotor Representation

Consider a point $X_w \in \mathbb{R}^4$ in homogenous coordinates corresponding to the 3D location of a point in the world coordinate frame. It is transformed by the extrinsics matrix $T_{cw} \in SE(3)$ and projected to the homogenous pixel coordinates $p_c \in \mathbb{R}^3$ in the image according to an invertible intrinsics matrix $K \in \mathbb{R}^{3\times3}$ that encodes the focal length and center pixel. The resulting well known relationship is

$$p_c = \frac{1}{X_c^z} \begin{bmatrix} K & 0 \end{bmatrix} T_{cw} X_w. \tag{3.1}$$

Where X_c^z is the third element of the product $T_{cw}X_w$. Without loss of generality, we assume that $K = I$.

Next, consider the warp function as known in computer vision. It has the property that

$$p_c(t) = W(t, t_0, p_c(t_0)). \tag{3.2}$$

That is, given the initial position of a world point in an image at time t_0, the warp function returns the position of the same world point in the image at another time. Note that warp functions are sometimes assumed to be linear, and this can be a good approximation when the region of interest is on a flat plane [42]. However, in general, the warp is nonlinear because its estimation

amounts to the integration of optical flow or tracking visual features such as points, lines, or patches.

Similarly to the warp function, the "flow map" from control theory, $\Phi_{f,u}$, is defined as returning the solution to a system defined by f with input u as follows:

$$\begin{aligned} \dot{x} &= f(x,u), \quad x(t_0) \in \mathbb{R}^n \\ x(t) &= \Phi_{f,u}(t, t_0, x(t_0)) \end{aligned} \tag{3.3}$$

Comparing (3.2) to (3.3) reveals that W and Φ serve the same purpose. W is the solution map giving trajectories in image space, whose derivative is also called optical-flow, and Φ is the solution map for the system state, whose time derivative is driven by the system dynamics. This similarity in purpose leads to an equality constraint.

Due to W's close relationship to position X through (3.1), many visual representations allow computing the position of the camera up to a characteristic scale such as the size of an object under fixation, the initial distance to a visual feature, or the baseline between two stereo cameras. We call the position to scale Φ_W and assume it can be estimated. In particular, the prominent families of computer vision algorithms, such as homography estimators, structure from motion, SLAM, and visual odometry result in such a Φ_W.

Φ_W can be related to position by a multiplicative scalar, that is the characteristic scale of the visual representation, which we call d. If the first three elements of the state are the position of a point in the camera's frame, i.e., $x = \begin{bmatrix} X_c^T & * \end{bmatrix}^T$, then Φ_W, d, and $\Phi_{f,u}$ are related by,

$$X_c(t) = \Phi_W(t, t_0, p_c(t_0))d = \Phi_{f,u}^{1,2,3}(t, t_0, x(t_0)), \tag{3.4}$$

where $p_c(t_0) = x^{1,2}(t_0)/x^3(t_0)$ and superscripts (i.e., $\Phi_{f,u}^{1,2,3}$) are used to denote the individual components of a vector. Note that this formulation holds for loose and tight couplings of vision and control, where loose coupling means the vision algorithm functions independently without the knowledge of action.

The problem of visuomotor control thus becomes finding specific forms of $\Phi_{f,u}$ and W that possess desirable properties. In particular, we seek a transformation that reformulates the problem into a form akin to those used in linear system theory, enabling us to take advantage of the properties of linear systems. Suppose the camera is translating but not rotating. Then the world frame's axes can be assumed to align with the camera frame's axis. In practice, this assumption holds for durations ranging from a few to several seconds when an inexpensive Inertial Measurement Unit is mounted in the same frame as the camera sensor and is used to compensate the image so that it can be considered as coming from a camera with fixed orientation. Consequently, the rotation between the camera frame c and the world frame w can be neglected as the frames can be assumed to coincide up to a translation and so X can be used in the place of X_c.

Then Newton's second law of motion can be applied in the fixed camera frame (within the time interval that the system can be considered rotation invariant). Let that time interval be $[t_0, t_0 + T]$ where $T \neq 0$. Then we obtain a linear system that relates the position, velocity, and acceleration components of the state X and Φ_W,

$$\Phi_W(t, t_0)d = X(t_0) + (t - t_0)\dot{X}(t_0) + \int_{t_0}^{t}\int_{t_0}^{\sigma} f_{mech}(X(\sigma_2), \dot{X}(\sigma_2), x_{act}(\sigma_2), u(\sigma_2), \theta)d\sigma_2 d\sigma. \tag{3.5}$$

Here f_{mech} encapsulates the mechanical dynamics due to forces exerted by actuators, θ is a set of mechanical parameters (arm lengths, mass, etc.) that can be considered constant, $\dot{X}$ is the time derivative of X, and x_{act} is the state of the actuator dynamics. We encapsulate the actuator dynamics as a separate system

$$\dot{x}_{act}(t) = f_{act}(x_{act}(t), u(t), \theta_{act}), \tag{3.6}$$

where θ_{act} are the constant parameters of the actuators.

Consider this visuomotor representation as known by an embodied agent without calibration to an external scale. Then X is the robot's 3D position relative to the point of interest in an unknown unit, d is the characteristic scale of vision in the unknown unit, Φ_W is the unitless estimate of the position due to vision, θ and θ_{act} are the constant parameters of the mechanical system and the actuators in unknown units. Finally, x_{act} and u are the state of the actuators in unknown units.

The only quantities known in general by an uncalibrated embodied agent system are then Φ_W and u. But, because Φ_W is unitless, if the embodiment attempts to estimate X, d, θ, and θ_{act} so that (3.5) and (3.6) hold at all times, the units of the estimated quantities must be implicitly determined by u. For example, it can be seen that the distance units of X are determined by the distance units defined by the acceleration, that is, a distance unit over seconds squared,

whose magnitude is determined from the embodied signal u transformed into acceleration by the system's f_{act} and f_{mech}. In other words, the magnitude of X, $\dot{X}$, and d is determined by the magnitude of the the action u and thus their units are implied by u.

Consequently, the general path for employing Embodied Visuomotor Representation involves the following steps:

- Define a model that represents the physical dynamics of the agent. Both classical and learning-based models can be used.

- Ensure all quantities in the model arising from physical processes are unit-free or up to scale. For example, if gravity is to be modeled, it can be assumed the effect is non-zero, but the magnitude must not be assumed.

- Choose a visual representation Φ_W where the characteristic scale is related to a quantity of interest. For instance, in a grasping task, it may be advantageous to choose Φ_W so that d corresponds to the object's size. Similarly, d might be the initial distance to a tracked feature, line, or object in a navigation task.

- Define a constraint between the unknowns d, X, $\dot{X}$, θ, θ_{act}, and x_{act} that enables a unique and task-relevant representation to be estimated. In what follows, we give two specific formulations.

- Construct an estimator for the chosen representation of d, X, $\dot{X}$, θ, θ_{act}, and x_{act} and incorporate these estimates into the task execution.

3.2.2 Closed Loop Control with Embodied Visuomotor Representation

In what follows, we give two specific examples of constructing an estimator and using the resulting state for closed-loop control. In the first example, distance is estimated in units of action for a single input, single output system. In the second example, the multiple input case is considered. In that case, distance is most directly available in multiples of the characteristic scale of vision but can easily be converted to the units of action of any of the inputs. In both cases, stable closed-loop control is a consequence of the embodied representation.

For each example, we begin by constructing a model, a sliding window estimator, and a closed-loop controller which will have guaranteed stability under mild assumptions. The approach is a form of indirect-adaptive control because, first, a forward model is identified using our framework. Then, a controller is synthesized from that model [67].

3.2.2.1 Double integrator with unknown gain

The simplest system that can be represented in the framework of Embodied Visuomotor Representation is a robot moving along the optical axis while fixating on a nearby object. The dynamics are assumed to be that of a double integrator and the image provides the position up to the characteristic scale of vision. This system was also considered in [68], however, the exposition was limited. The system is given by

$$\dot{x} = \underbrace{\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}}_{A} x + \begin{bmatrix} 0 \\ b \end{bmatrix} u \tag{3.7}$$

$$\Phi_W(t, t_0)d = x_1(t). \tag{3.8}$$

Here, the observation (output) is Φ_W, and it is linearly related to the state x_1 through the unknown d. b is the unknown gain between control effort and acceleration in an external scale, such as meters per second squared. b implicitly depends on both the strength of the embodied agent's actuators and the mass of the embodiment itself. Dimensionality analysis reveals that b is simply the unknown conversion factor between the embodied scale implied by u and the external scale. Finally, the system is observable as long as $d \neq 0$. That is, the visual representation's characteristic scale, such as the initial distance to a visual feature, size of a patch, or the baseline between two stereo cameras, is non-zero.

A sliding window estimator that considers Φ_W, and u to be known over the interval $[t_0, t_0 + T]$, $T \neq 0$ results in the problem

$$\min_{d, x(t_0), b} \int_{t_0}^{t_0+T} \left(\Phi_W(t, t_0)d - x_1(t_0) - (t - t_0)x_2(t_0) - b \int_{t_0}^{t} \int_{t_0}^{\sigma} u(\sigma_2) d\sigma_2 d\sigma \right)^2 dt, \tag{3.9}$$

which cannot be solved uniquely without additional constraints because a trivial solution can be realized by allowing all optimized variables to equal zero. However, dividing by b, the unknown conversion factor between external scale and embodied scale, reveals a problem that can be solved

as long as acceleration is non-zero for some period during the interval $[t_0, t_0 + T]$

$$\min_{\frac{[d,x(t_0)]}{b}} \int_{t_0}^{t_0+T} \left(\Phi_W(t, t_0)\frac{d}{b} - \frac{x_1(t_0)}{b} - (t - t_0)\frac{x_2(t_0)}{b} - \int_{t_0}^{t}\int_{t_0}^{\sigma} u(\sigma_2)d\sigma_2 d\sigma \right)^2 dt. \quad (3.10)$$

Since b is simply a conversion factor between units, we see that the lumped quantities d/b, $x_1(t_0)/b$, and $x_2(t_0)/b$ are traditional state estimates but in the embodied units of u. Thus, we see that an embodied agent can naturally estimate state in an embodied scale by comparing what is observed with the accelerations effected through action.

In practical applications, Φ_W will be noisy because it is estimated using an image made out of discrete pixels. Even if this noise is zero mean, Equation (3.10) results in a biased estimator because it uses Φ_W as an independent variable. An unbiased estimator can be realized by dividing by d/b, resulting in the problem,

$$\min_{\frac{[b,x(t_0)]}{d}} \int_{t_0}^{t_0+T} \left(\Phi_W(t, t_0) - \frac{x_1(t_0)}{d} - (t - t_0)\frac{x_2(t_0)}{d} - \frac{b}{d}\int_{t_0}^{t}\int_{t_0}^{\sigma} u(\sigma_2)d\sigma_2 d\sigma \right)^2 dt. \quad (3.11)$$

In the new problem, Φ_W is the dependent variable because it is no longer multiplied with one of the optimized variables. Then, because the problem is linear least squares, if Φ_W is corrupted with zero mean noise, the parameter estimates will remain unbiased. On the other hand, the estimated parameters have changed. The state x is now estimated in multiples of d instead of the desired multiples of b. Further, only the reciprocal of the d/b is estimated. These estimates can be transformed back to the desired quantities with division, which will result in some bias since

division does not commute with expectation. However, unlike when using Equation (3.10), techniques that increase the accuracy of estimates, such as taking more measurements, will increase the accuracy of the transformed estimates. A detailed explanation is provided in the Supplemental Material.

The solution to either formulation is unique as long as the acceleration, u, is non-zero for a finite period within the sliding window. A detailed proof is given in the Supplemental Material.

Then if the control law $u = K(x/b)$ is applied, the closed loop system dynamics become

$$\dot{x} = Ax - \begin{bmatrix} 0 \\ \\ b \end{bmatrix} K\frac{x}{b} = Ax - \begin{bmatrix} 0 \\ \\ 1 \end{bmatrix} Kx. \tag{3.12}$$

Because the unknown gain b cancels out in the rightmost expression, we can say that the closed loop behavior of the system is invariant to the value of b. That is, while b is not known individually, and is only a term in the known ratios x/b and d/b, it will not affect the systems behavior. Thus, the control gains K can be chosen as if $b = 1$ and x itself is known. This is despite the fact that b appears individually in the dynamics model. Similarly, the characteristic scale of vision, d, does not affect the closed loop behavior despite remaining unknown.

Further, consider that once d/b is known, $\Phi_W(t)d/b$ provides a direct estimate of $x_1(t)$ in the embodied unit. Thus, in theory, it is sufficient to solve (3.10) or (3.11) once, and subsequently treat control of (3.7) as a traditional output feedback problem. Finally, since the closed loop system is linear, time-invariant, controllable, and has known parameters, feedback gains K always exist and can be chosen to ensure global closed-loop stability. In particular, since this system corresponds to a PD controller applied to a double integrator, the stabilizing gains can be easily chosen.

3.2.2.2 Unknown actuator dynamics

Next, the previous example is extended to the case that the robot does not know its actuator dynamics. Typically, actuator dynamics produce a "lagging" response to control inputs. For example, they could take the form of an unknown delay or low-pass filter between the control input and its effect on acceleration. Additionally, the full 3D motion is considered.

Suppose the dynamics are fully linear, the robot can move in three dimensions, there is an unknown, stable linear system between control inputs and the 3D acceleration, and a visual process estimates the position of the robot up to the characteristic scale of vision. Then, the dynamics can be expressed as

$$\dot{x} = \frac{d}{dt}\begin{bmatrix} X \\ \dot{X} \\ x_{act} \end{bmatrix} = \begin{bmatrix} \dot{X} \\ Bu + C_{act}x_{act} \\ A_{act}x_{act} + B_{act}u \end{bmatrix} \tag{3.13}$$

$$\Phi_W(t, t_0)d = X$$

Estimating the unknown parameters B, A_{act}, B_{act}, and C_{act} simplifies to satisfying the following equality constraint

$$\begin{aligned} &\Phi_W(t, t_0)d = X(t_0) + t\dot{X}(t_0) \\ &+ \int_{t_0}^{t}\int_{t_0}^{\sigma} Bu(\sigma_2) + C_{act}\left[\Phi_{A_{act}}(\sigma_2, t_0)x_{act}(t_0) + \int_{t_0}^{\sigma_2}\Phi_{A_{act}}(\tau, t_0)B_{act}u(\tau)d\tau\right]d\sigma_2 d\sigma \end{aligned} \tag{3.14}$$

Suppose that the actuator dynamics expressed inside the double integral are BIBO stable. Then, the impulse responses relating the inputs of the actuator's system to this term go to zero exponentially fast. Consequently, it is sufficient in many practical applications to approximate these dynamics as a finite length convolution with the input. Then, we get the following equality constraint where G is a matrix value signal whose i,j'th entry is to be convolved with the j'th element of u to get its effect on the i'th state.

$$\Phi_W(t, t_0)d = X(t_0) + (t - t_0)\dot{X}(t_0) + \int_{t_0}^{t}\int_{t_0}^{\sigma}(G * u)(\sigma_2)d\sigma_2 d\sigma. \tag{3.15}$$

As with the previous example considering a double integrator, any problem based on this constraint has no unique solution because every term has an unknown multiplier. However, unlike the double integrator, we cannot simply divide by b to get a problem with a unique solution. Instead, we can divide by the scalar d to get a linear constraint that will be uniquely satisfied given sufficient excitation by u. A detailed proof is given in the Supplemental Material. Consider the sliding window estimator again. The full problem to be solved is

$$\min_{\frac{[X(t_0),\dot{X}(t_0),G]}{d}} \int_{t_0}^{t_0+T}\left[\Phi(t, t_0) - \frac{X(t_0)}{d} - (t - t_0)\frac{\dot{X}(t_0)}{d} - \left(\frac{G}{d} * \int_{t_0}^{\cdot}\int_{t_0}^{\sigma} u(\sigma)d\sigma_2 d\sigma d\sigma\right)(t)\right]^2 dt, \tag{3.16}$$

where the fact that the convolution with G can be brought out of the double integral has been used, and $\cdot$ at the top of the double integrator is the placeholder for the variable that will be

convolved over.

In this case, the position of the agent is recovered in multiples of the characteristic scale of vision due to division by d. However, the units of distance can still be recovered in the embodied units of acceleration by dividing all estimated parameters by the gain of one of the actuator's impulse responses at a particular frequency. In particular, if the DC gain of the actuator dynamics is non-zero, we can consider that

$$\left(\frac{X(t_0)}{d}\right) / \left(\frac{\int_0^\infty g_{ij}(t)dt}{d}\right) = \frac{X(t_0)}{\int_0^\infty g_{ij}(t)dt}, \tag{3.17}$$

Where g_{ij} is the i, j entry of G and $\int_0^\infty g_{ij}(t)dt$ is its scalar valued DC gain. Thus, distance in units of action is still available. Further, the distance in the units of action is different for each input, and the conversion factor between all the embodied scales is known to the agent. So, it is free to switch between units as may be convenient.

If the actuator dynamics are minimum phase, and thus an inverse impulse response G^{-1} exists, then since G/d is known, dG^{-1} can be determined. It is then straightforward to synthesize a reference tracking controller. Consider the control scheme given by

$$\begin{aligned} e &\coloneqq \begin{bmatrix} X_{ref} \\ \dot{X}_{ref} \end{bmatrix} - \begin{bmatrix} X/d \\ \dot{X}/d \end{bmatrix} \\ u &\coloneqq dG^{-1} * Ke. \end{aligned} \tag{3.18}$$

Then the closed-loop dynamics are

$$\ddot{X} = G * u = G * (dG^{-1} * Ke) = K \left(d \begin{bmatrix} X_{ref} \\ \dot{X}_{ref} \end{bmatrix} - \begin{bmatrix} X \\ \dot{X} \end{bmatrix} \right), \tag{3.19}$$

where the fact that the systems G and dG^{-1} cancel (by definition) except for the scale d has been used.

The resulting closed loop's stability does not depend on the magnitude of the embodied scale or any external scale. Instead, the reference point is in multiples of vision's characteristic scale, which could be the size of a tracked object, the baseline between a stereo pair, or the initial distance to an object. As before, it is straightforward to convert to the embodied units of action because d can be replaced with any actuators DC gain, $\int_0^\infty g_{ij}(t)dt$. In that case, the closed loop dynamics become

$$\ddot{X} = K \left(\int_0^\infty g_{ij}(t)dt \begin{bmatrix} X_{ref} \\ \dot{X}_{ref} \end{bmatrix} - \begin{bmatrix} X \\ \dot{X} \end{bmatrix} \right). \tag{3.20}$$

As with (3.12), the systems in (3.19) and (3.20) are linear, time-invariant, controllable, and all parameters are known. Thus, gains K always exist and can be selected to ensure closed loop stability [41].

In practice, the inverse of the actuator dynamics should not be used in the control law because the cancellation of dynamics typically results in a control law with poor performance and robustness. Regardless, we consider the inverse dynamics above so that the final close loop

control law clearly illustrates the units being employed by Embodied Visuomotor Representation. However, in general, it is now straightforward to design traditional control laws using the identified parameters. In particular, the jumping experiments avoid using inverse dynamics.

3.2.3 Applications

We now turn to three basic robotic capabilities: touching, clearing, and jumping, which can be accomplished by uncalibrated robots that use Embodied Visuomotor Representation.

3.2.3.1 Uncalibrated Touching

Consider an uncalibrated robot with a monocular camera that must touch a target object in front of it. The contact will occur at a non-zero speed because the robot does not know its body size and thus cannot come to a stop just as it reaches the target. Further, the robot also does not know the strength of its actuators, the size of the target, or the size of anything else in the world. In what follows, we carefully apply the five steps for using Embodied Visuomotor Representation to design an algorithm for this problem. Figure 3.2 outlines the resulting uncalibrated touching procedure visually.

Starting with the first step, we assume a double integrator system as in (3.7), that is the robot accelerates along the optical axis according to $\ddot{x}_1 = bu$ where $\ddot{x}_1$ is acceleration in meters per second squared and u is a three-dimensional control input. Upon inspection, we see that the only quantity defined by the world, b, does not have a predetermined unit. Thus, the second step is complete.

To complete the third step, assume the robot is facing a planar target. The reciprocal of

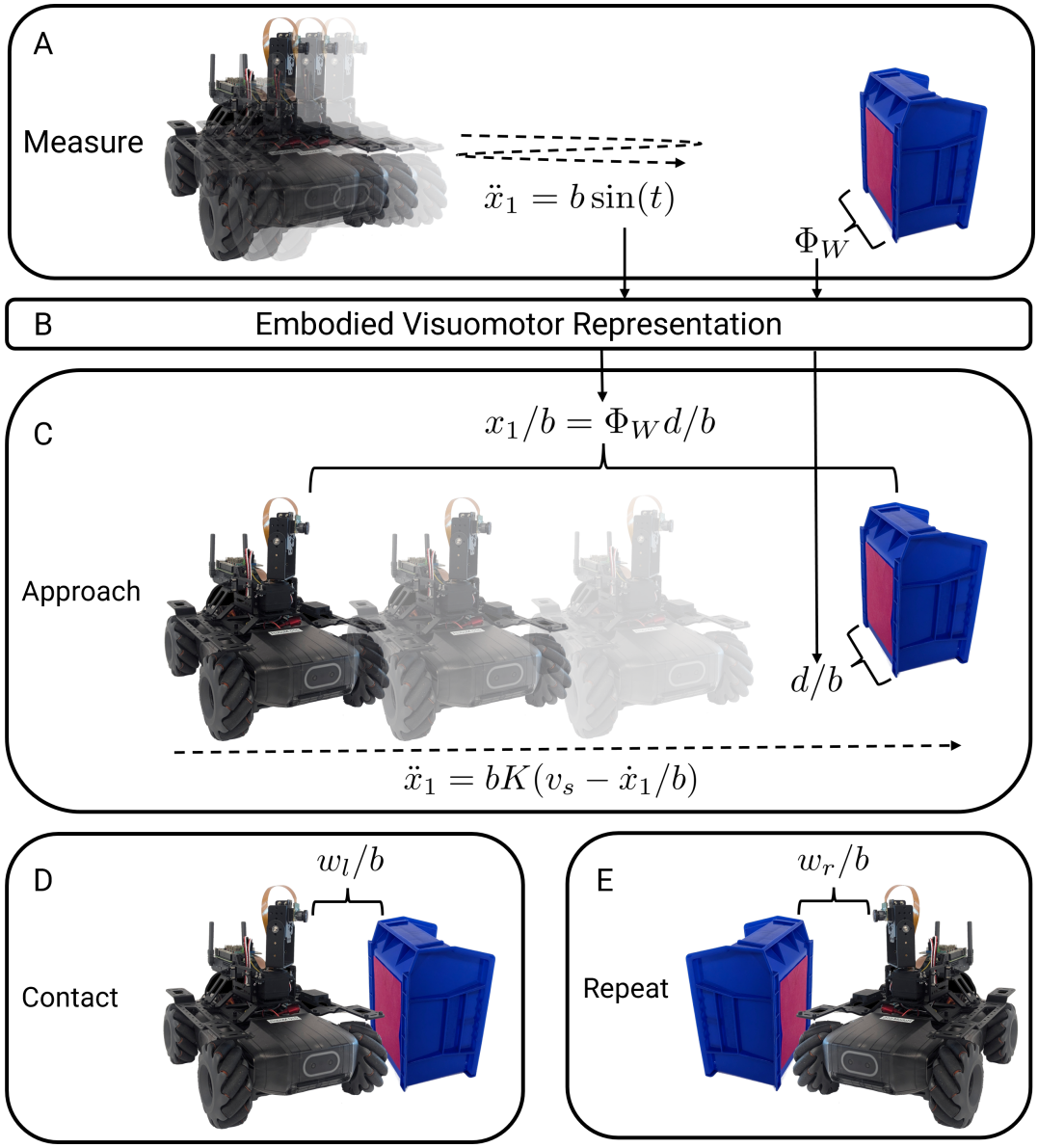

Figure 3.2: **Overview of the procedure for uncalibrated touching using Embodied Visual Representation.** (A): The robot oscillates by applying an open loop control input $\sin(t)$ while measuring the size of the touching target in the visual field as Φ_W. (B): Embodied Visuomotor Representation uses the control input and visual information to estimate x_1/b and d/b, which are the size of and distance to the target in the embodied unit. (C): With the size of the target known, it is possible to approach the target at a desired safe contact speed, v_s, using closed-loop control. In the closed loop, the effect of the unknown embodied gain b cancels except for its interaction with the setpoint v_s. (D): Upon making contact with the target, the distance between the camera and the target is the size of the body w_l in the embodied unit. (E): The procedure can be repeated, by approaching from different directions, to approximate the convex hull of the robot. A supplementary movie illustrating the procedure is available at `https://prg.cs.umd.edu/EVR`.

the target's width in normalized pixels within the visual field can then be used as Φ_W. That is, $\Phi_W = Z/d$, where Φ_W's characteristic scale is the unknown width of the target, d. It should be noted that this simple visual representation is valid only for flat objects that are coplanar with the camera, making it and following experiment pedagogical in nature. However, in general, (3.4) demonstrates that Embodied Visuomotor Representation is agnostic to the particular visual representation or algorithm used so long as it provides position up to a characteristic scale.

The fourth step, which involves establishing a constraint between vision and control that includes the unknowns, is accomplished by recalling Equation (3.8) and dividing through by b. This results in the unknowns $x(t_0)/b$ and d/b, which correspond to the initial conditions, $x(t_0)$, and the characteristic scale of vision, d, in the embodied units of u. Once this is done, the fifth step can be applied: estimating the unknowns using Equation (3.11).

The unique estimate of d/b obtained from this formulation allows the distance between the robot and the target to be estimated at all times via the relation $\hat{x}_1/b = \Phi_W(t)(d/b)$. This can be combined with the target's position in the image to recover the target's 3D position. Closed-loop control can then use this distance to center the robot on and approach the target along a direction in the camera frame at a constant speed, because in the closed loop, the remaining unknown, b, cancels out, as shown in Equation (3.12).

To ensure safe contact without damaging the robot, the designer must specify a safe contact speed, v_s, in embodied units. This limitation mirrors biological systems, where animals often create safe environments for their young to prevent injury, ensuring that the maximum achievable speed does not cause harm. It is important to note that the robot cannot stop just before touching the target, as it does not know how where its body is relative to the camera.

In practice, the robot's designer or a higher-level cognitive process can set v_s based on mul-

tiples of the target's dimensions or by considering multiples of the maximum force (in embodied units) that the robot should experience upon collision. An example of this calculation is provided in the Supplemental Material. Specifically, if the designer knows the size of the target in meters and a safe speed in meters per second, they can convert the safe speed to multiples of the target's size per second, which the robot can subsequently interpret in embodied units.

Subsequently, contact with the object can be detected either with a touch sensor or by using the visual position estimate $\Phi_W d$ to detect that the robot has stopped getting closer to the target. Thus, the robot can approach a target at a constant safe speed in a given direction in the camera field and touch it, regardless of the embodiment and environment's specifications.

It should be noted that it was assumed that the open-loop oscillations would not cause the robot to come in contact with the obstacles. However, this is not strictly necessary. Equation (3.11) has a unique solution given any acceleration of arbitrarily small duration. Thus, in theory, it is always possible to estimate the unknowns and switch to closed loop control prior to reaching the target.

Figure 3.2 visually outlines the uncalibrated touching procedure. Movie 1, which is linked to in the caption, details the experimental procedure for both uncalibrated touching and clearing. Figure 3.3 shows the estimated width, distance, and velocity over time during the measurement phase of uncalibrated touching for three different control input gains, $b = 0.5$, 1, and 2. Since the conversion from embodied units to meters is determined by b, the embodied estimate is expected to align with the ground truth only when $b = 1$. Similarly, Figure 3.4 shows estimated distances and control inputs as the robot approaches and touches the target while using the same three ground truth values for b. The approach velocity was fixed to the same embodied value for each trial, meaning when $b = 0.5$, the robot takes twice as long to reach the target compared to $b =$

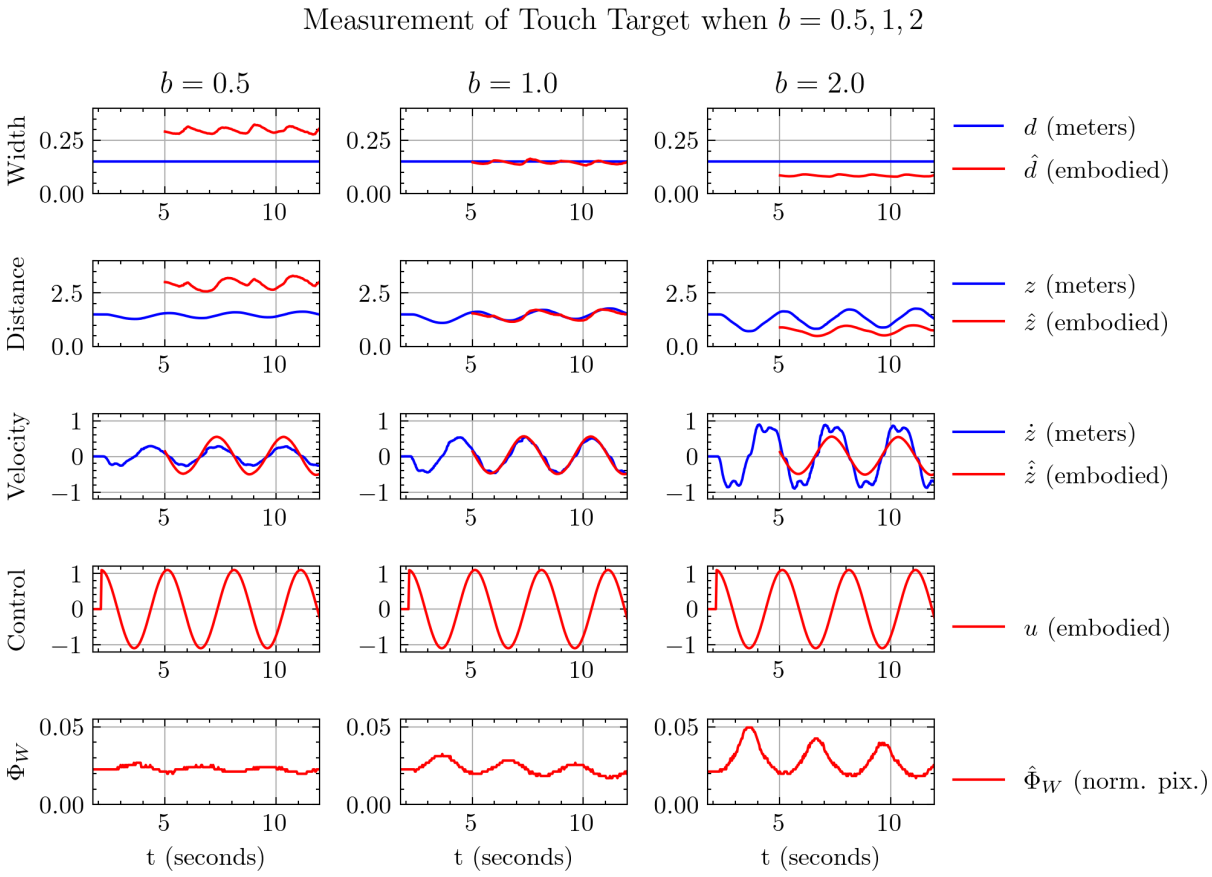

Figure 3.3: Experimental data and estimated signals from the target measurement phase of uncalibrated touching for three different values of the control gain b. The first, second, and third rows of plots show the estimated target width, distance from the target, and velocity as estimated by the sliding window formulation in Equation (3.12). The bottom two rows show the inputs to the sliding window formulation, that is, the control signal u and the width of the target in the visual field in normalized pixel coordinates Φ_W. As expected, when $b = 1.0$, the embodied estimates are closely aligned with the ground truth distances measured in meters. However, when $b = 0.5$ or $b = 2.0$, the embodied estimates are twice as much and half, respectively, of the ground truth value in meters.

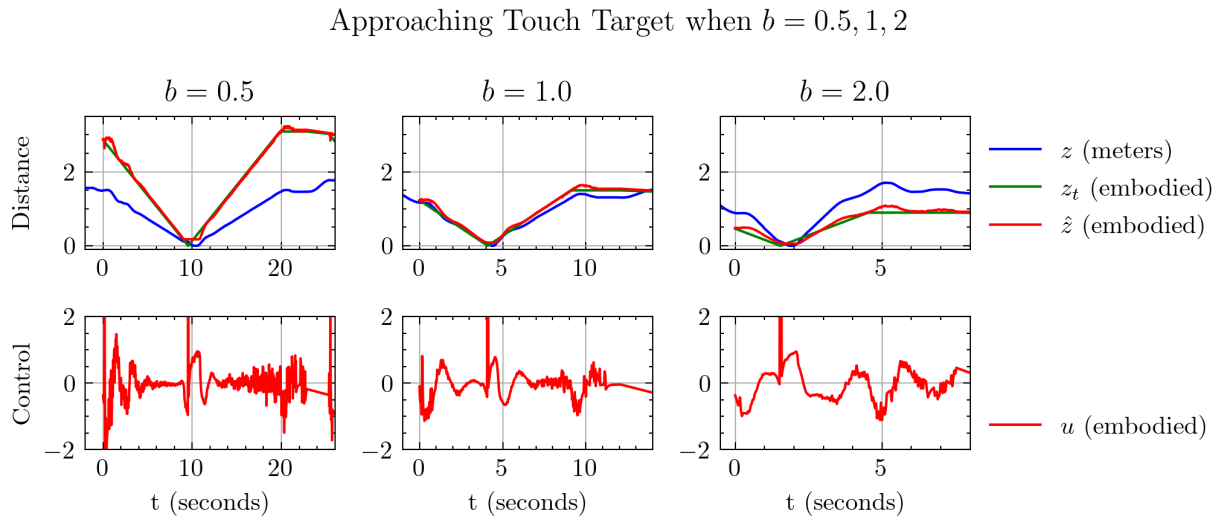

Figure 3.4: Experimental data and estimated signals from the target approach phase of uncalibrated touching for three different values of the control gain b. The methods take different amounts of time to reach the target because the approach speed was specified as a fixed value in the embodied unit while b was varied. Note that the closed loop behavior remains stable and qualitatively similar despite the control input gain varying by a factor of 4. As shown by Equation (3.12) this is expected.

1. Whereas when $b = 2$, it takes half the time. Alternatively, the methods discussed above for specifying a safe contact speed allow the robot's designer to ensure a consistent approach speed regardless of b.

Additionally, two experiments were conducted to validate the accuracy of estimated embodied representations. In the first, the uncalibrated touching procedure was performed with the robot starting from an initial distance of 50, 75, 100, 150, 225, and 350 cm away from a 15 centimeter wide touch target. At each distance, the input gain b was set to 0.5, 1, and 2. When $b = 1$ and $b = 2$, the minimum initial distances were 75 cm and 150 cm, respectively, to avoid hitting the target during the open loop oscillation. In total, 70 trials were run. 350 cm was the maximum testable distance; beyond this, the procedure failed due to the limited width of the target in the field of view (<6 pixels). Figure 3.5 shows the resulting embodied target widths and embodied robot widths compared to ground truth. Table 3.1 quantifies the error in Figure 3.5 as

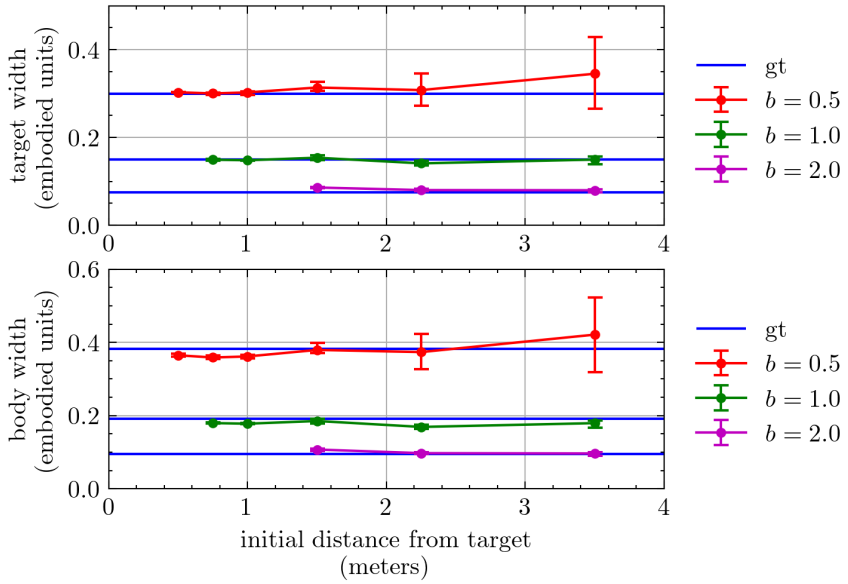

Figure 3.5: Measured width of touching target and width of the robot's body, in embodied units, as the initial distance from the target and the control input gain b are varied. Error bars show mean, minimum, and maximum value over five trials.

a percentage of the ground truth target and body width.

In the second experiment, the uncalibrated touching procedure was performed with the robot initially 150 cm from targets of ground truth width 3.75, 7.5, and 15 cm and input gain b equal to 0.5, 1, and 2. Five trials of each parameter combination were run, resulting in a total of 45 trials. Figure 3.9 shows the resulting embodied target widths and embodied robot widths compared to ground truth. The ground truth widths result from converting the width of the robot in meters minus the turning radius of the pan-tilt mount to the embodied distance. Table 3.2 quantifies the error in Figure 3.9 as a percentage of the ground truth target and body width.

In Figures 3.5 and 3.9, the blue ground truth lines result from converting the ground truth width of the target and width of the robot to meters. The plotted ground truth robot width was reduced by the turning radius of the pan-tilt mount that the camera is mounted on in embodied units.

	Initial Distance (cm)					
	50	75	100	150	225	350
Control Gain	**Target Width Error (%)**					
$b = 0.5$	1.00	0.61	1.37	4.62	7.42	20.80
$b = 1.0$	-	0.66	1.17	3.05	5.91	3.61
$b = 2.0$	-	-	-	14.87	7.00	6.53
	Body Width Error (%)					
$b = 0.5$	4.51	6.08	5.33	2.33	7.19	21.28
$b = 1.0$	-	6.00	6.90	3.16	11.52	6.08
$b = 2.0$	-	-	-	12.30	2.01	3.66

Table 3.1: Mean absolute error in percent of the estimated target width (15 cm) and the observable robot body's width (19.1 cm) as the initial distance from the target is increased. Each configuration was tested over 5 trials. The true robot body's width is 25.4 cm. However, the observable width is reduced by the pan-tilt camera's turning radius. At 350 cm, the target occupied 6 pixels in the field of view.

During all trials, the robot applied an open-loop 1/3 Hz acceleration signal, resulting in positional oscillation with 25 cm of amplitude when $b = 1$. The sliding window estimator was configured to consider 10 seconds of data.

3.2.3.2 Uncalibrated Clearing

Once a robot can touch, it can quickly determine if it can clear obstacles of initially unknown size prior to reaching them as shown in Figure 3.6 and in Movie 1. In the following, the first four steps of and the estimation portion of step five are the same as in the touching application. Thus, we focus on the last portion of the fifth step, which is applying quantities estimated via Embodied Visuomotor Representation in the task.

Let the robot approach the touching target along direction v in the body frame; then, if the body extends further than the camera in the direction of v, the robot's body will touch the target prior to the camera reaching the target. Let the time of contact be called t_c. At the time of

	Target Width (cm)			Opening Width (cm)		
	3.75	7.5	15	12.5	25	37.5
Control Gain	**Target Width Error (%)**			**Opening Width Error (%)**		
$b = 0.5$	13.00	11.45	5.45	8.22	2.21	3.16
$b = 1.0$	5.25	6.91	1.10	7.03	2.91	2.20
$b = 2.0$	14.79	36.85	13.15	16.99	14.96	17.52
	Body Width Error (%)					
$b = 0.5$	19.88	4.74	4.14			
$b = 1.0$	21.39	8.98	5.69			
$b = 2.0$	12.57	20.66	10.07			

Table 3.2: Mean absolute error in percent of the estimated touch target or opening width and observable robot body's width (19.1 cm) as the target or opening width is varied. Each configuration was tested over 5 trials. The true robot body's width is 25.4 cm. However, the observable width is reduced by the pan-tilt camera's turning radius. The initial distance was set to 150 cm. Only the touch target trials are associated with an estimated body size.

contact, the vector from the camera to the point on the body that is touching the target is known and corresponds with a point on the perimeter of the robot's body. Repeating the touching process by approaching a target from many directions allows the robot to approximate its convex hull in the scale of the embodied units.

Clearing obstacles is now as simple as measuring the distance from the robot's camera to the object in the embodied unit and comparing that to the robot's size in the embodied unit. In particular, as illustrated by Figure 3.2, if a robot has measured its width in the embodied units as $w/b = w_l/b + w_r/b$, where w_l/b and w_r/b are distance from the camera to the left and right side of the robot respectively, then it only needs to measure width of the opening, as illustrated in Figure 3.6, and compare it to the width of the robot. Let the positions of the left and right sides of the opening be X_l and X_r; then the robot measures their positions as X_l/b and X_r/b. Thus, the robot fits if

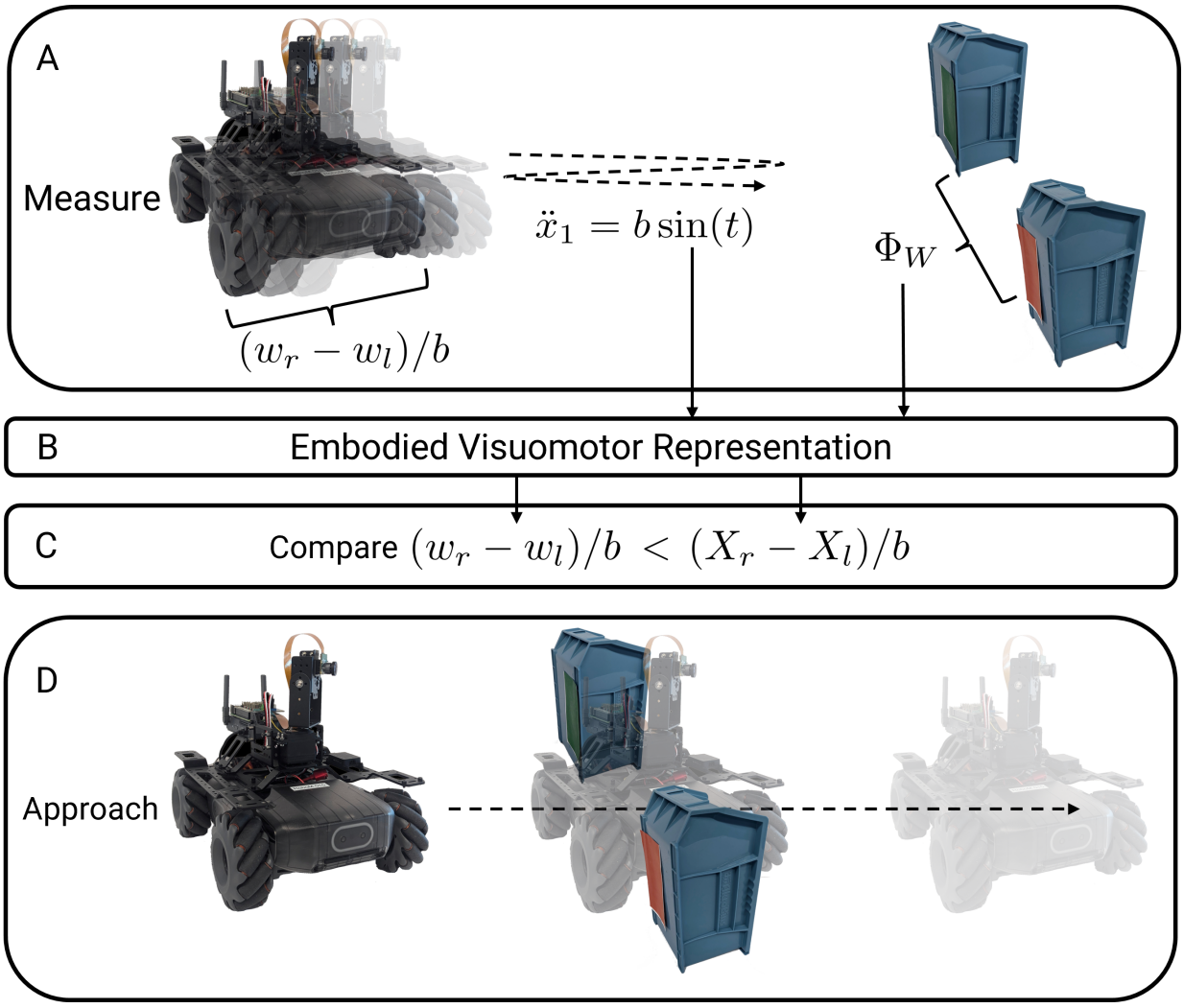

Figure 3.6: **Overview of the procedure for clearing obstacles using Embodied Visual Representation.** (A): The robot oscillates by applying the control input $\sin(t)$ in open loop input while using the size of the opening between two obstacles in the visual field to estimate its distance to scale, Φ_W. (B): Embodied Visuomotor Representation uses the control input and visual information to estimate the opening size, $(X_r - X_l)/b$, in the embodied units. (C): The size of the robot in the embodied unit, $(w_r - w_l)/b$, as estimated with a series of uncalibrated touches, is compared to the size of the opening in the embodied unit to determine if the robot can fit or clear the opening. (D) If the robot can fit, the known size of the opening can be used to determine the position of the robot, and closed-loop control can guide the robot through the opening.

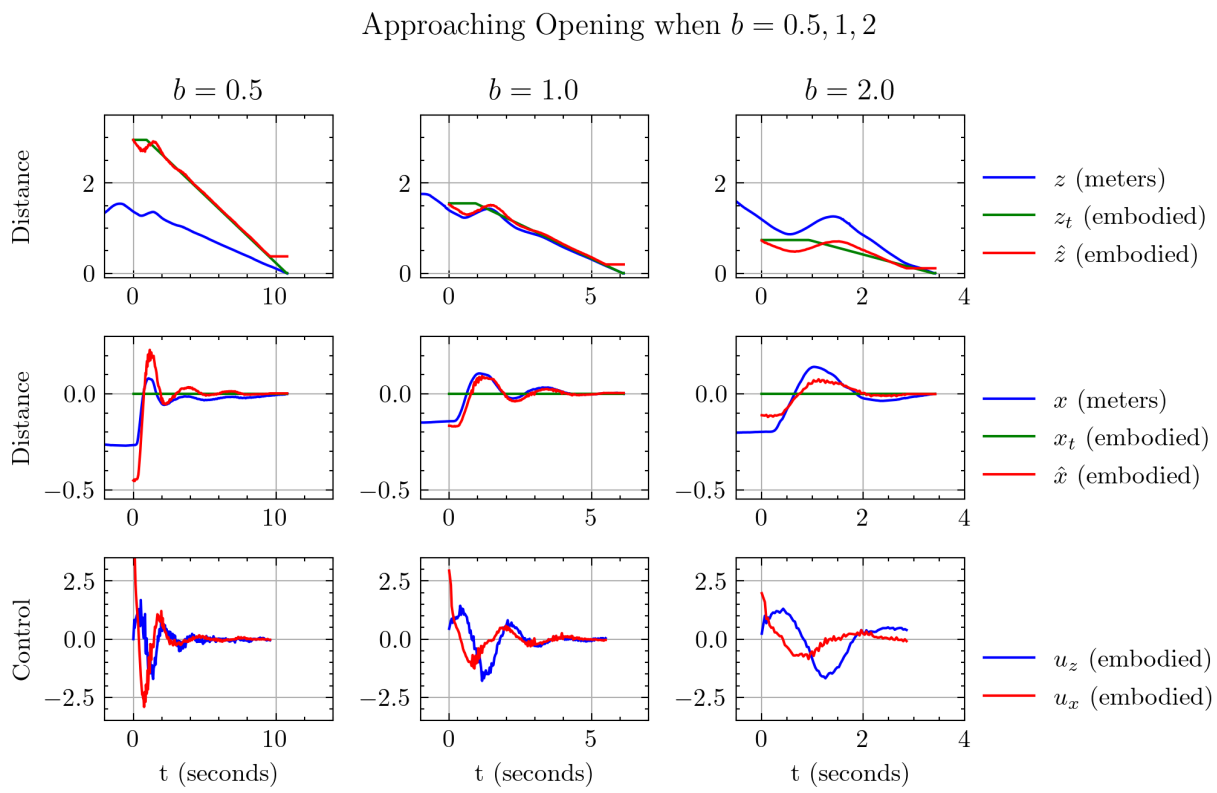

Figure 3.7: Experimental data and estimated signals from the opening approach phase of uncalibrated clearing for three different values of the control gain b. The methods take different amounts of time to reach the target because the approach speed was specified as a fixed value in the embodied unit while b was varied. Note that the closed loop behavior remains stable and qualitatively similar despite the control input gain varying by a factor of 4. As shown by Equation (3.12), this is expected.

$$w < \|X_l - X_r\| \iff \frac{w}{b} < \left\| \frac{X_l}{b} - \frac{X_r}{b} \right\|. \tag{3.21}$$

If the robot fits, it can proceed through the opening using the same controller that was used to touch targets.

Figure 3.7 shows estimated distances from the opening as the robot approaches using different ground truth values for $b = 0.5$, 1, and 2 as before. The approach corrects a translational perturbation in the X axis in addition to approaching the target at a constant speed along the Z axis. The approach velocity was fixed to the same embodied value for each trial; thus, when $b =$

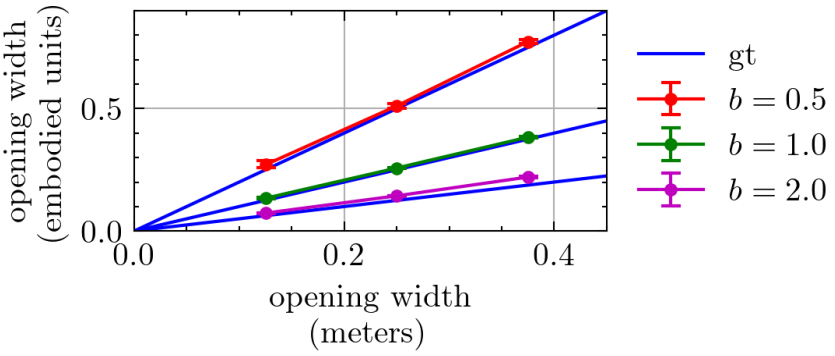

Figure 3.8: Measured width of opening width, in embodied units, as the width of the opening and the control input gain b are varied. Error bars show mean, minimum, and maximum value over five trials.

0.5 and $b = 2$, the robot takes twice and half as long, respectively, to reach the target as when $b = 1$.

Additionally, Figure 3.8 shows the measured width of the opening as a function of the opening's true width. Openings of size 12.5, 25, and 37.5 cm were tested with the robot's initial distance from the opening fixed at 150 cm. Each configuration of opening size and control gain b were tested 5 times for a total of 45 trials. It should be noted that measuring an opening's width is substantially similar to measuring a target's width, and so Figure 3.8 can be interpreted as an extension of the upper plot in Figure 3.9. Table 3.2 quantifies the error in Figure 3.8 as a percentage of the width of the opening.

3.2.3.3 Uncalibrated Jumping

Suppose an uncalibrated robot with two legs, that apply force with an unknown first order response (or lag), and a monocular camera needs to jump a gap to a platform an unknown distance away but at the same height. In what follows, we carefully apply the five steps for using Embodied Visuomotor Representation. Figure 3.10 and Movie 2 of the Supplemental Material outline the

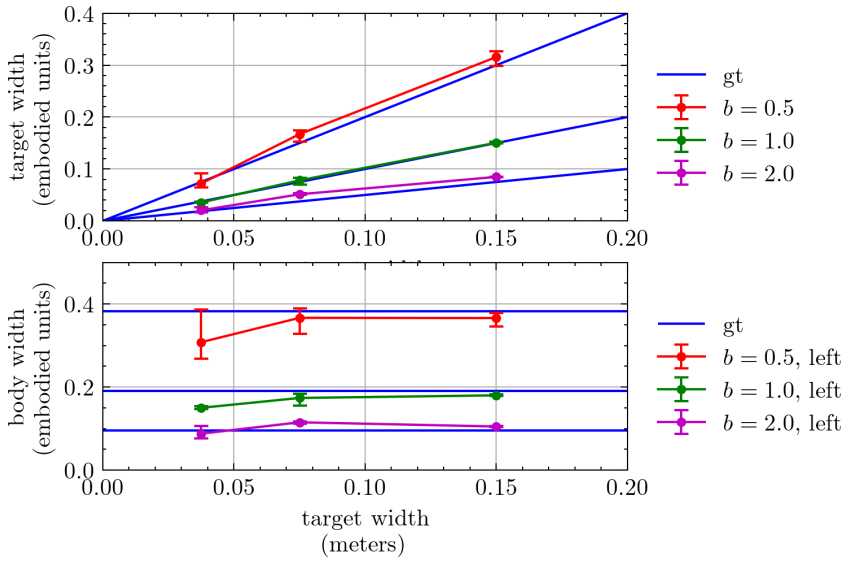

Figure 3.9: Measured width of touching target and width of the robot's body, in embodied units, as the width of the target and the control input gain b are varied. Error bars show mean, minimum, and maximum value over five trials.

uncalibrated jumping procedure visually. A link to the movie is provided in the figure caption.

Starting with the first step, we assume a classical model of the system. Let the legs applied force respond to inputs according to a first order low pass filter with a time constant $1/\alpha, \quad \alpha > 0$. That is,

$$\begin{aligned} \ddot{X}_2 &= b x_{act} - g_b \\ \dot{x}_{act} &= \alpha \left(u - x_{act} \right), \end{aligned} \tag{3.22}$$

where $\ddot{X}_2$ is acceleration in the vertical direction and g_b is an unknown gravitational bias. Then, as with the the touching example, all the quantities defined by the world, that is b, g_b, and α do not have predetermined units or magnitudes. Thus, the second step is completed.

To complete the third step, assume the robot fixates on a horizontal line that it needs to

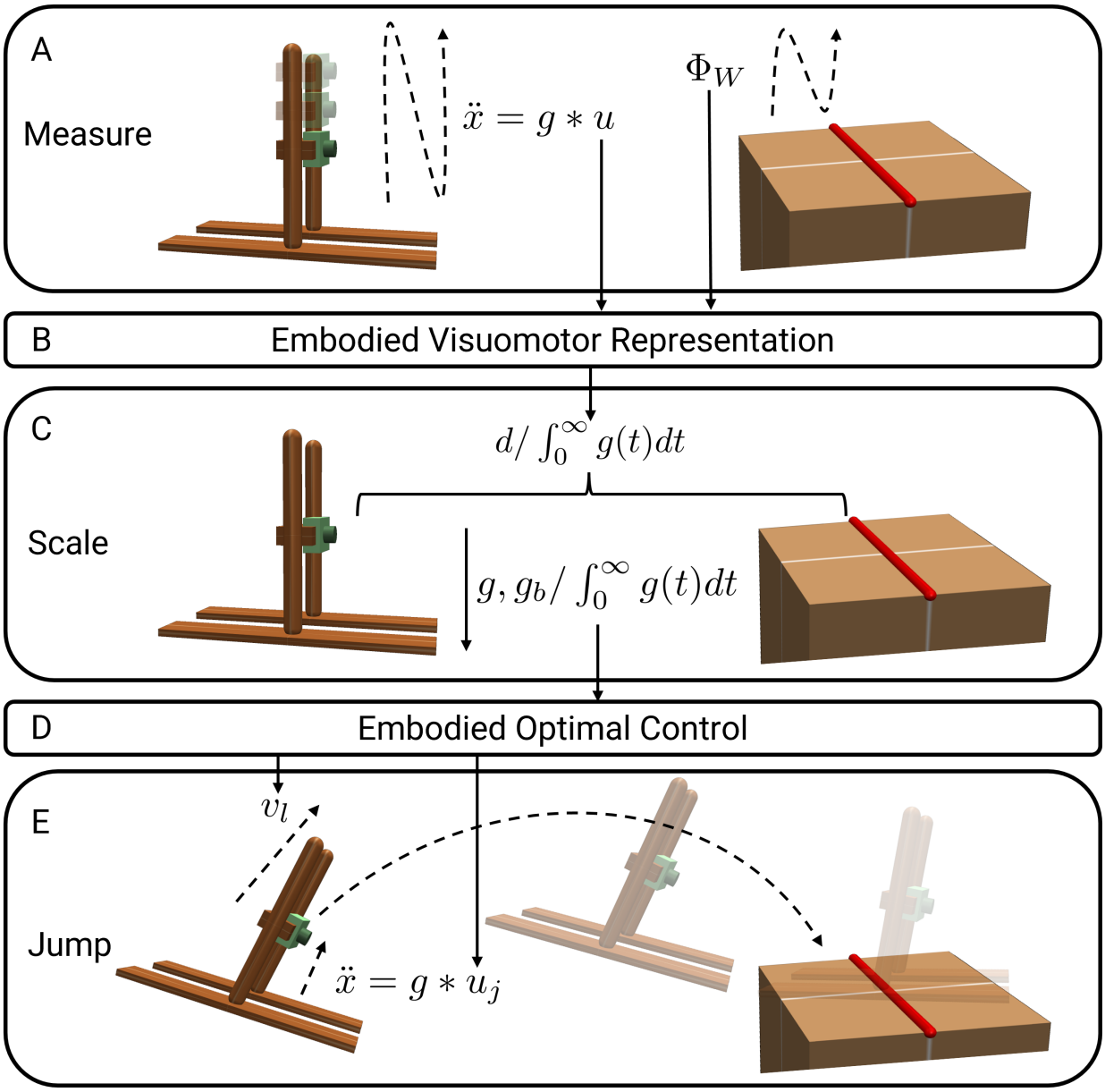

Figure 3.10: **Overview of the procedure for jumping a gap using Embodied Visual Representation.** (A): The robot oscillates up and down using a high gain control while measuring the control input u and vertical position in the visual field Φ_W of a target line (red). The vertical acceleration achieved $\ddot{X}_2$ results from filtering the control input by the unknown actuator dynamics g. (B), (C): Embodied Visuomotor Representation uses the control input and visual information to estimate the actuator dynamics g, distance to the jumping target d, and strength of gravity g_b in the embodied units $\left(\int_0^\infty g(t)dt\right)^{-1}$. (D): An embodied optimal control problem is solved for the control input u_j of minimum total variation that will reach the required launch velocity v_l to jump the gap. (E) The jumping control input is executed in open loop after tilting the body forward. A supplementary movie illustrating the procedure is available at `https://prg.cs.umd.edu/EVR`.

jump to and oscillates up and down. Then, the line's height in the image in normalized pixels can be used as Φ_W. That is, $\Phi_W = X_2/X_3$ where X_2 is the vertical position of the line and X_3 is the distance between the camera and the line. Then, the characteristic scale of vision d is equal to X_3 and is the distance to be jumped.

The fourth step of establishing a constraint between vision and control that involves the unknowns is accomplished by recalling Equation (3.14), specializing it to this example's dynamics, and adding the additional gravitational bias term. Finally, the fifth step can applied: estimating the force of gravity and the distance to the line in embodied units using Equation (3.16). The full formulation is given in the methods section and specifically Equation (3.23). There, a family of truncated exponential decay functions are used to represent the first order actuator dynamics, which results in a linear least squares problem and a unique solution. Subsequently, Equation (3.17) can be used to recover the distance to be jumped in the embodied units as $\hat{X}_3(t_0)/b$. Figure 3.10 illustrates this procedure.

Then, given the embodied jumping distance d, embodied gravity estimate, and a predetermined launch angle θ_l, the robot can solve the projectile motion equations to determine a launch velocity, v_l, and the landing time t_f. To find control inputs that achieve this launch velocity, we pose a convex optimal control problem, (3.25), that solves for a non-negative control input with minimum total variation that reaches the launch velocity. The formulation of both problems is given in the methods section.

Figure 3.11 illustrates the results of this procedure in terms of the achieved jumping trajectory (in meters) and the applied control signal u in embodied units. Figure 3.12 illustrates the accuracy of the estimate of the distance to be jumped, d, and the gravitational force, g_b, in the embodied unit after completing the measurement phase. Estimates resulting from processing

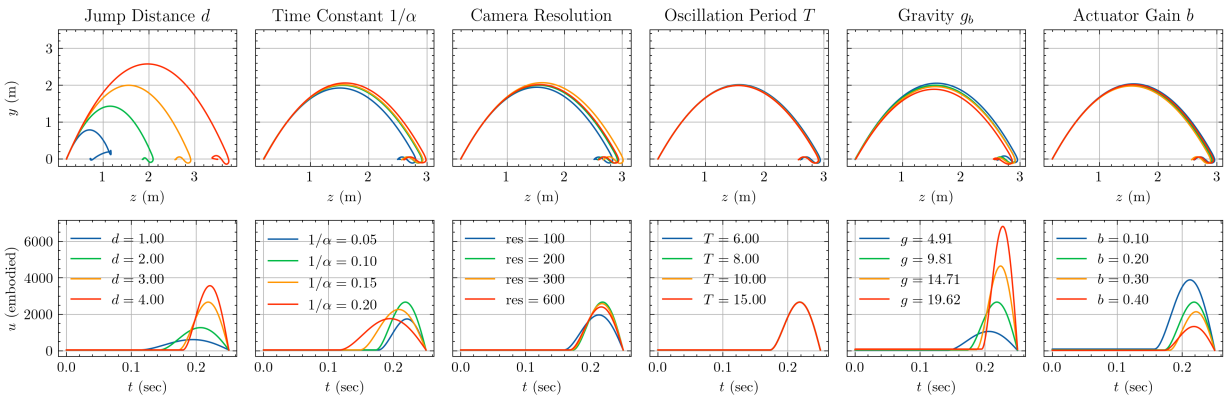

Figure 3.11: **Jump trajectories and jump control signal versus experimental parameters.** The procedure can successfully jump over a wide range of unknown gaps varying from 1 to 4 meters in width. A nominal gap of 3 meters can be jumped despite variations in the unknown actuator time constant, camera resolution, oscillation period, gravitational force, and actuator gain. The jump control signals resulting from an embodied optimal control problem (3.25) can be seen to vary in magnitude and duration as expected given varied jump distance, actuator time constant, gravitational strength, and actuator gain.

simulated images with color thresholding and simulating worst case pixel quantization errors are shown. The error due to finite resolution can be seen to decrease by approximately $1/\text{resolution}$. A theoretical justification for this is given in the Supplemental Material. For each set of experiments, the non-varied parameter values were set to $d = 3.0$, $1/\alpha = 0.1$, $\text{res} = 200$, $T = 10$, $g_b = 9.81$, and $b = 0.2$.

It should be noted that performing the up and down oscillation requires a controller to compensate for the effects of gravity. For this experiment, the control gains were chosen manually, but they could easily be chosen automatically using traditional system identification and control design methods. Additionally, the mass of the robot's legs and feet have been neglected as the effect of their mass on the jumping trajectory cannot be estimated by oscillating the body alone. For the purpose of this paper, it suffices to assume the mass of the legs and feet is small, and so does not significantly affect the achieved jumping trajectory. However, the effect of the added mass of the legs and feet can be estimated by doing a vertical test jump prior to jumping the gap.

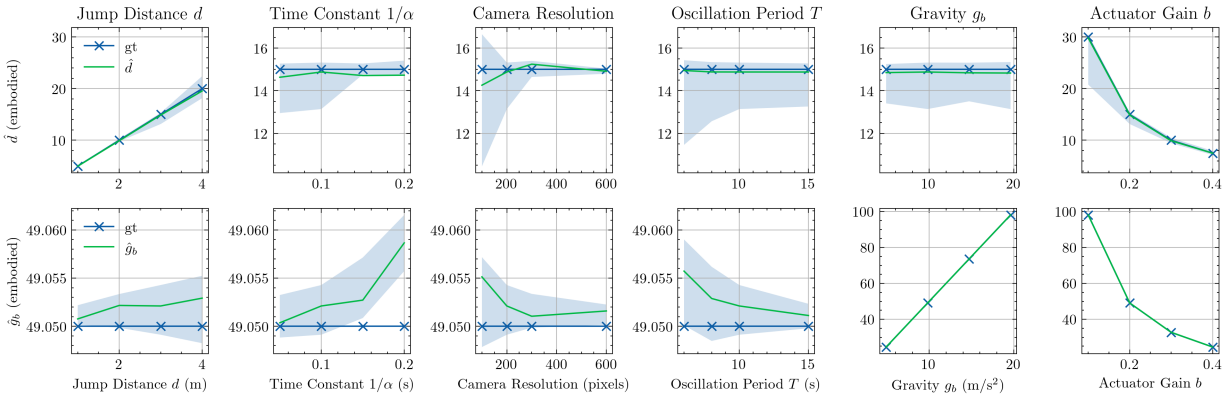

Figure 3.12: **Estimated jump distance and gravitational strength in embodied units versus experimental parameters.** The blue ground truth corresponds to ground truth values. Green estimates result from using an image and color thresholding to estimate Φ_W. Blue shaded regions illustrate the maximum error achieved via simulated pixel quantization errors varied over 26 trials. Larger jump distances are estimated less accurately. Lowering the camera resolution below 200 pixels increases estimation error quickly. Increasing the oscillation period slightly improves results. Lower actuator gain, which results in less oscillation, decreases the accuracy of the estimated distance. While some variation in the estimate of gravitational bias can be observed, the error is typically on the order of 0.02 %. The effects of varying the actuator time constant and gravitational strength are included for completeness when comparing to Figure 3.11.

Finally, because the approach is model based, it, in theory, works for all gap sizes, gravitational strengths, and actuator time constants. However, given fixed parameters, it will fail in specific scenarios where the model's assumptions do not hold. For example, if the strength of gravity is made very weak, the agent may fall off the jumping platform during the measurement phase unless the oscillation frequency and magnitude are also reduced.

3.2.4 Comparisons to Bees and Gerbils

Next, we contrast the properties of algorithms using Embodied Visual Scale with behavior observed in bees and gerbils.

Bees have been recently shown to understand the size of an opening in wingbeats during their approach to an opening. It is thought that the size of the wings is learned during early de-

velopment through contact with the hive. Additionally, the bees oscillate increasingly vigorously as openings are made smaller, seemingly to determine the size of the hole more accurately [14]. Similarly, our method for uncalibrated clearing determines the size of the robot in embodied units through contact and oscillates to determine the size of the opening. Additionally, an examination of the least squares formulation admits that increased oscillation over longer periods will increase the accuracy and robustness of the estimated quantities because errors can be averaged out.

Similarly, Mongolian Gerbils vibrate their body up and down before making a jump over a gap of unknown distance. It is thought that this helps their visuomotor system determine the appropriate muscle stimulation to clear the gap [15]. Like bees, the intensity of the oscillations increases when the gap is larger and thus is more difficult to jump over. Our approach also requires oscillation, can estimate the appropriate actuator stimulation to jump a gap, benefits from larger oscillations (which induces a larger baseline for vision), and can increase its accuracy by considering longer timer intervals.

3.3 Discussion

Embodied Visuomotor Representation is a concrete methodology that allows robots to emulate the process of estimating distance through action – a capability well established in the natural sciences. This approach simplifies implementation by allowing low-level control laws to self-tune and removing the need for detailed physical models or calibrated 3D sensors. As a result, even uncalibrated robots can reliably perform tasks such as touching, clearing obstacles, or jumping, while maintaining guaranteed stability.

Traditionally, body-length units have been proposed as a natural basis for embodied scale.

However, such measurements are not directly accessible to an uncalibrated robot. Embodied Visuomotor Representation addresses this limitation by providing a mechanism through which a robot can internally estimate or re-estimate body-length units as needed. Instead of relying on external units like meters, it grounds the perception of distance using the robot's own acceleration response to control inputs.

We further compare traditional approaches based on sense-plan-act with our approach based on Embodied Visuomotor Representation in Table 3.3. The table compares the main algorithmic components of traditional approaches that might be used to achieve touching, clearing, and jumping. This comparison reveals how the stability of traditional approaches becomes implicitly tied to prior knowledge of the conversion factor between an embodied scale and an external scale due to the assumption of control inputs on an external scale. Embodied Visuomotor Representation can escape this situation by using a different architecture from the traditional sense-plan-act cycle as illustrated by Figure 3.1.

The development of Embodied Visuomotor Representation has borrowed ideas broadly from many areas, in particular the Tau Theory, Glenberg's theory of embodiment as memory, the Internal Model Principle, and Visual Self-Models. Below, these frameworks are compared and contrasted to Embodied Visuomotor Representation. We further compare our framework to leading approaches in robotics, including Direct Optical Flow Regulation, Visual Servoing, Visual Inertial Odometry (VIO), Visual Self-Modeling, and Deep Visuomotor Control.

Task	External Scale	Embodied Visuomotor Representation (EVR)
Touching	State estimated with calibrated sensors in external scale	Embodied state estimated with EVR
	Control stabilizes state via conversion to embodied signals	Control stabilizes embodied state with embodied signals
	Stability depends on prior knowledge of conversion factor	Stability is invariant to units
Clearing	Robot's physical size defined in external scale	Embodied physical size estimated with touch
	State estimated with calibrated sensors in external scale	Embodied state estimated with EVR
	The clearing algorithm uses a world in an external scale and clears objects via conversion to embodied signals	The clearing algorithm uses an embodied world that is consistent with embodied signals
	Stability depends on conversion and a pre-determined physical model	Stability is invariant to units and physical dimensions
Jumping	Jumping distance estimated using calibrated sensors	Jumping distance and actuator dynamics estimated with EVR
	Launch velocity estimated in external scale	Launch control signal estimated via embodied representation
	Control achieves launch velocity via conversion to embodied signals	Launch control signal applied open loop
	Success dependent on scale and pre-determined controller	Stability invariant to units and controller

Table 3.3: Robot behaviors that can be accomplished with external scale and traditional methods, but can also be accomplished with Embodied Visuomotor Representation (EVR). Due to its use of embodied scale, Embodied Visuomotor Representation theoretically guarantees success and stability without calibrated sensors, pre-determined physical models, and low-level controllers that are tuned prior to deployment.

3.3.1 Comparisons to existing frameworks

3.3.1.1 Tau Theory (Time-to-contact)

The *Tau Theory* originally proposed an embodied visuomotor relationship as the basis for how time-to-contact could be used by humans for tasks such as driving a car [13]. This was developed into the *General Tau Theory* [27], which aims to explain numerous behaviors using time-to-contact (tau). Later theories generalized it to motion in all directions, called looming [69]. Tau Theory does not provide a specific methodology for achieving tau trajectories and so roboticists have proposed methods ranging from direct regulation [28, 29, 59] to indirect estimation of position followed by regulation [30, 68, 70], with most relying on pre-tuned of control laws to account for the unknown embodied scale. Embodied Visuomotor Representation can use time-to-contact as the basis for a scaleless visual transition matrix Φ_W as in [68] without committing to any particular control law. Further, it is easy to use the framework to verify if a proposed control law is stable.

3.3.1.2 Direct Optical Flow Regulation

A generalization of Tau Theory is *Direct Optical Flow Regulation*. Here, arbitrary optical flow fields are used as the setpoint to a control loop. Such control schemes have been shown to result in striking similarities to insect flight behavior [56, 71–73]. Notably, [60] showed that the framework allows orientation with respect to gravity to be estimated using oscillations of orientation. Such schemes depend on pre-tuning of the control law for the expected heights and velocities. [31] addresses this limitation by lowering the open-loop gain when oscillations occur.

Embodied Visuomotor Representation can continuously estimate embodied distance without necessarily oscillating. Thus, it can be used to follow more general trajectories while simultaneously scaling the gains of a direct optical flow regulator to ensure stability.

3.3.1.3 Visual Servoing

Another approach to visuomotor control is *Visual Servoing* which considers objectives based on position (pose based) or image coordinates (image based) [57, 58]. When combined with Embodied Visuomotor Representation, pose based approaches no longer require a 3D model of known scale as demonstrated by uncalibrated touching and clearing. This situation is similar to that considered by Image Based Visual Servoing, which operates without knowledge of scale by basing the control objective on the pixel positions and mitigating the effects of unknown depth using approximations, partitioning, or adaptivity [57, 58, 74]. Embodied Visuomotor Representation provides an estimate of depth, and so is closely related to adaptive visual servoing. However, Embodied Visuomotor Representation is also a broader framework that enables a rich set behaviors such as the examples of touching, clearing, and jumping.

3.3.1.4 Memory as Embodiment

Our implementation of these behaviors has strong parallels with *Glenberg's theory of embodiment as memory* [63]. Glenberg writes, "embodied representations do not need to be mapped onto the world to become meaningful because they arise from the world" and our proposed Embodied Visuomotor Representation does just that. A core tenet of Glenberg's theory is the "merge" operation that allows memories of different types of actions to be combined to do a

new task. Our Embodied Visuomotor Representation admits a path towards realizing a general motion-based "merge" operator because it explicitly represents the system's dynamics.

3.3.1.5 Internal Model Principle

Realizing such advanced behavior requires the ability to predict. This is supported by the *Internal Model Principle*, that suggests an internal model of the process to be regulated must be contained within a controller if it is to succeed. This principle has analogs in psychology, where it is accepted that humans depend on internal models to control themselves [65, 75]. Further, recent evidence strongly suggests that even simple animals such as dragonflies have such predictive visuomotor models [55, 66]. In computer science, *Animate Vision* was proposed to explain how such visuomotor control processes can be learned through interaction with the environment [8, 48]. These theories are very general and do not explicitly consider the challenges introduced by the unit-less nature of vision. Thus, Embodied Visuomotor Representation can be seen as a specialization that can realize versions of these theories in robots.

3.3.1.6 Visual Self-Modeling

An important component of predictive modeling is understanding the shape of the body. While the given examples of uncalibrated touching and clearing are the most basic way to do this, it is of interest to combine Embodied Visuomotor Representation and recent work on *Visual Self-Modeling* that consider neural representations of a robot's 3D form [76].

3.3.1.7 Deep Visuomotor Control

Similarly, it is of interest to understand Embodied Visuomotor Representations application to *Deep Visuomotor Control* as exemplified by [77]. Recent work has considered visual navigation, where a robot must learn to navigate through a scene from visual inputs and output concrete actions to be taken by low-level controllers provided by a variety of morphologies [78–80].

3.3.1.8 Visual Inertial Odometry

Next, we turn to *Visual Inertial Odometry* (VIO), and its relation to Embodied Visuomotor Representation. VIO is the conventional approach to estimating a robot's state from vision and calibrated inertial measurement units (IMUs). Typical approaches follow a match, estimate, and predict architecture to estimate a metric trajectory and the 3D position of tracked points [24, 25, 35, 46]. Architecturally, this is identical to sense-plan-act but in the estimation setting. In contrast, Embodied Visuomotor Representation generally considers the characteristic scale of vision, which may not be the distance to features. This encourages an object-centered visual framework where control tasks are considered with respect to individual objects. Similarly, the role of IMUs is different within Embodied Visuomotor Representation. They are useful for estimatating rotation over short periods. and their acceleration can be used up to scale to better estimate mechanical parameters. They also allow a robot to continuously estimate the conversion factor between its embodied scale and an external scale as might be needed to adhere to a safety standard or follow high level directions.

3.3.2 Future Work

The above comparisons to existing concepts resulted in several directions for future work. However, there are many more, and so below, we outline the future work in theory and applications that we consider most important.

3.3.2.1 Theory

In terms of theory, this paper has considered only equality-constrained Embodied Visuomotor Representation. That is, the unitless quantities from vision are considered to exactly equal some operator of control inputs. Ideas from psychology, and, in particular, the Tau Theory, suggest that more qualitative processes will suffice for many applications. Thus, it is of interest to develop inequality-constrained Embodied Visuomotor Representation in hopes of realizing qualitative representations and qualitative control laws that robots can use to accomplish tasks. Similarly, it is of interest to develop Embodied Visuomotor Representation in combination with direct methods such as Direct Optical Flow Regulation, and Image Based Visual Servoing. Here, an embodied representation of distance can be used to help initialize, learn, or select direct control laws. Finally, the role of rotation has recently been found to be useful for estimating attitude from direct measurements [60]. Thus, it is of interest to carefully consider the role of rotation in embodied representation.

Turning back to equality-constrained Embodied Visuomotor Representation, we have presented two types of estimation problems. The first, exemplified by (3.10), estimates quantities in embodied units of action. The second formulation is given by (3.11) and (3.16) which estimate quantities in units of the visual processes characteristic scale. Subsequently, these quantities can

be converted to an embodied unit of action. While the latter were shown to be unbiased with respect to zero mean noise in Φ_W, it is still of interest to perform a more general and careful error analysis with the goal of constructing robust or optimal estimators for Embodied Visuomotor Representation.

Next, we discuss the representation of dynamics. The derivation of Embodied Visuomotor Representations above neglected damping and spring effects, as might be caused by friction or tendon-like attachments, respectively. However, these effects can be incorporated, assuming these naturally occurring dynamics are stable, which they almost always will be. For example, we can incorporate into an Embodied Visuomotor Representation a system's dissipating energy due to spring and damping effects by considering velocity or position as the fundamental quantities controlled by the action instead of acceleration. In this paper, acceleration was used as the fundamental quantity controlled through action because all animals and robots must use forces.

Finally, it is interesting to consider more general dynamics than linear systems and how to control them via Embodied Visuomotor Representation. In principle, except for the use of unit-less visual feedback, this is not substantially different from existing work in robot dynamics, and we expect many techniques to port over directly. On the other hand, the linear dynamics considered in this paper should be sufficient for many applications. Thus, it is of interest to apply recent advancements in adaptive control, such as kernel methods that guarantee estimation of BIBO stable impulse responses [81].

3.3.2.2 Applications

Next we consider future work in applications. A straightforward application that can be accomplished via Embodied Visuomotor Representation is avoiding and pursuing dynamic objects. If a position-based approach is used, the basic procedure is identical to the touching, clearing, and jumping, except that an additional model of the dynamic object must be considered. Catching is also of interest. However, it is thought that humans use direct methods (that is, they do not primarily rely on 3D position) to catch objects. The evidence is particularly strong in sports where humans achieve extraordinary performance despite sensorimotor delays [82, 83]. Then, since Embodied Visuomotor Representation allows for learning predictive self models that could be used for delay compensation (as is thought to be essential in biological systems [55, 65]), it is of particular interest to study the theoretical connections between embodied representation and direct methods in the task of robot catching.

We also consider imitation and embodied affordance estimation to be the next two most important applications. Assuming the embodied mechanical and dynamics parameters have already been estimated, imitation within Embodied Visuomotor Representation becomes a problem of determining the embodied control signals that accomplish a visually observed behavior. The characteristic scale-based representation of vision will then naturally allow robots to imitate an action using the units of their own embodiment instead of an external scale.

Finally, while this paper has exclusively considered visual processes and their connection to motor signals, the basic principles should be extended to consider multiple modalities such as tactile sensing and audio. We are particularly interested in tactile sensing because grasping is an action that results in both tactile and visual responses.

3.4 Methods

3.4.1 Hardware

The robot platform used for the touching and clearing experiments is a DJI RoboMaster EP. The platform features mecanum wheels, which allow it to translate omnidirectionally. The provided robot arm and camera were removed and replaced with a pan-tilt servo mount and a Raspberry Pi Camera Module 3 with a 120 degree wide-angle lens running at 1536×864 resolution and 30 frames per second. A Raspberry Pi 5 interpreted images and calculated control commands for the robot. The associated code is provided in the Supplemental Material.

The DJI RoboMaster does not support direct acceleration control. The lowest level supported is velocity control. To emulate the double integrator described in the touching and clearing experiments, the acceleration commands from the proposed control laws were transformed into velocity commands using a leaky integrator with a time constant of 10 seconds.

3.4.2 Uncalibrated Touching

The touching experiment uses color thresholding to detect the pixels of a planar target and calculate Φ_W based on the assumption of a flat target parallel with the image plane. As mentioned in the results section, this visual representation was chosen as a pedagogical example and will not work for non-flat objects. In general, the theory holds without modification for any other vision algorithm that provides position to scale as per (3.4). The robot oscillates in open-loop by applying a 1/3 Hz sinusoidal acceleration signal over a 10 second interval. The amplitude of acceleration was set so that the positional oscillations had an amplitude of 25 cm

when $b = 1$. Equation (3.11) is used to solve for the current position, velocity, and size of the touching target in embodied units using a 3 second sliding window of measurements. To do so, the outermost integral of Equation (3.11) is approximated in discrete time, so that the problem can be considered as a standard linear least squares problem of the form $\min_x \|Ax-b\|^2$. This problem is transformed to the normal equations $A^T Ax = A^T b$ and x is computed using JAXopt's [84] Cholesky decomposition based solver for linear systems. The code is available in the supplemental material.

Subsequently, the robot drives toward the target at a constant velocity of 0.3 embodied units per second using PID control and positional feedback from vision in embodied units. The approach continues for the time required to traverse the initial distance between the robot and the target. Since the robot's body is closer to the target than the camera, the robot makes contact with the target sometime during this interval. Then, the minimum distance between the robot's camera and the target is the horizontal distance between the robot's camera and the edge of the robot that made contact with the target first.

The PID control loop was tuned manually and was accomplished with just a few trials because the closed-loop behavior does not depend on the units of distance or the geometry of the scene. Thus, reasonable initial values for the PID gains can be determined from a desired closed loop bandwidth.

3.4.3 Uncalibrated Clearing

The clearing experiments consider whether the robot can fit through an opening (clear it) prior to reaching the opening. To do so, the robot determines the width of the body in the

embodied unit by touching its left and right sides to a nearby object using the previously described procedure. It remains for the robot to measure the size of an opening in the embodied unit. Once again, the left and right sides of the opening are detected with color thresholding. The left and right edges of the right and left sides of the opening, respectively, are used as the characteristic scale of vision. As with the last example, this visual representation is pedagogical. The same sinusoidal acceleration and sliding window estimator used during uncalibrated touching then estimates the size of the opening in the embodied units. If the robot is smaller than the opening, it proceeds through with the same PID-based position control as used for uncalibrated touching.

3.4.4 Uncalibrated Jumping

The jumping robot experiment was performed in a MuJoCo simulation [85]. Codes for this simulation are provided in the Supplemental Material.

The robot consists of a body on two legs and feet. The legs are modeled as cylinders or pistons, with first-order activation dynamics and a time constant of 0.1 seconds. The body mass was set to 10 kg, and the mass of the legs and feet amounted to 0.22 kg. Two angular position controllers hold the legs upright during the measurement phase of the experiment and tilt the body forward approximately 20 degrees during the jumping phase.

The measurement phase of the jumping experiment consists of three parts. In the first, the minimum/maximum position of the body is estimated by recording the initial position as the bottom position, increasing the upwards force until the body begins to lift, initializing a PID controller's integrator term with the force that overcomes gravity, and using PID control to maintain a constant upwards velocity until the top position is reached. Subsequently, the PID

controller oscillates the body for 10 seconds at 1 Hz around the midpoint between the bottom and top positions with an amplitude equal to $1/4$ of the distance between the top and bottom positions. While this approach introduces an assumption of a pre-tuned PID controller to control the body position, this could be eliminated by applying traditional system identification techniques in the embodied units.

During oscillation, color thresholding is used to determine the vertical normalized pixel coordinate corresponding to a red line placed in the middle of the target platform. The characteristic scale of vision then becomes the distance between the robot and the red line. Subsequently, the measurements of the control effort applied by the PID control and the vertical normalized pixel coordinate are used to solve for the leg actuator's impulse response, the distance between the robot and the red line, and the initial velocity.

The simulated camera has a resolution of 200×200 pixels and a 90 degree vertical field of view and runs at 60 fps. Estimation of the maximum possible errors due to pixel quantization, as illustrated in Figure 3.12, was achieved by quantifying the ground truth pixel position of the red line to integer coordinates with random quantization boundaries according to the formula $p_{sim} = \mathrm{round}(p_{gt} + \Delta) - \Delta$ and varying the value of Δ between $[-0.5, 0.5]$ with 26 steps.

To approximate the actuator impulse response, a linear combination of 50 first-order impulse responses truncated to 4 seconds with a DC gain of 1 and time constants ranging from 0.008 to 0.5 seconds are considered. The resulting quadratic program to estimate the distance to the line, initial velocity, and actuator dynamics is then given by:

$$\min_{\frac{\dot{X}_2(0),g_b,c}{d}} \int_2^T \left(\Phi_W(t) - t\frac{\dot{X}_2(0)}{d} - \frac{1}{2}\frac{g_b}{d}t^2 - \frac{g}{d} * \int_0^t \int_0^\sigma u(\sigma_2) d\sigma_2 d\sigma \right)^2 dt$$

$$\frac{g(t)}{d} = \sum_i c_i e^{-t/\tau_i} \tag{3.23}$$

$$\text{s.t.} \quad c_i >= 0 \quad \forall i$$

where $\dot{X}_2(0)$ is the initial velocity on the robot camera's vertical axis, g_b is the gravitational bias force, c_i are the coefficients of the basis functions for g and τ_i are the corresponding time-constants of the basis functions. The double integral and convolution are computed using a 0.001 second approximation, corresponding with the simulation's timestep. The outermost integral is approximated with a $1/60$'th of a second timestep, corresponding to the 60 fps simulated camera. The loss is evaluated over the interval $[4, T]$ instead of $[0, T]$ to prevent initial conditions from contributing to the actuator dynamics through the 4 second long impulse response g. Non-negativity of c_i is necessary to prevent overfitting to numerical integration errors. Similarly to the uncalibrated clearing and touching implementation, the problem is solved by approximating the outermost integral in discrete time. This results in a constrained quadratic program, which is solved using CVXOPT's quadratic program solver [86]. The code is available in the supplemental material. By solving this problem, the dynamics of the system are estimated automatically.

To convert from units of the characteristic scale to embodied units of action, we divide each estimated quantity by $\int_0^\infty g(t)dt/d$. In what follows, all quantities can be assumed to be in this embodied unit.

Subsequently, the standard equations for projectile motion are combined with the estimated embodied distance and gravitational force to solve for an embodied launch velocity v_l that will

jump the gap. The equations to be solved are

$$\begin{aligned} 0 &= t_f \sin(\theta_l) v_l - \frac{1}{2} t_f^2 g \\ d &= t_f \cos(\theta_l) v_l, \end{aligned} \tag{3.24}$$

where t_f is the time of landing.

Finally, A terminally constrained optimal control problem is solved to find the control input with a minimum total variation that accelerates the body from rest to the launch velocity in 250 milliseconds over the distance oscillated over during the measurement phase. The resulting quadratic program is:

$$\begin{aligned} &\min_u \int_0^T \left(\frac{d}{dt} u(t) \right)^2 dt \\ \text{s.t.} &\int_0^T (g * u)(t) dt = v_l \\ &\int_0^T \int_0^\sigma (g * u)(\sigma_2) d\sigma_2 d\sigma = d_m \\ &u(0) = 0, \quad u(T) = 0, \quad u(t) >= 0, \quad \forall t \in [0, T] \end{aligned} \tag{3.25}$$

where v_l is the necessary launch velocity in embodied units, and d_m is the distance from the start position at which the launch velocity should be achieved. d_m is necessary to ensure the launch velocity is reached prior to reaching the maximum position of the body with respect to the legs.

Non-negativity of u is required to prevent lifting the lightweight legs from the jumping platform prior to reaching the launch velocity. We formulate and solve the problem using a discrete time approximation of 0.001 seconds. Because of exponential decay, $(g * u)(t) > 0$

when $t > T$, and so the leg actuators are deactivated for $t > T$ to prevent the body's velocity from exceeding v_l. The resulting embodied jump control signal u is offset by the estimated gravitational bias and executed in an open loop.

Thus, by combining (3.23) and (3.25), the estimation of system dynamics and the synthesis of the jumping control input are fully automated, except for the use of a PID controller during the oscillation phase. However, this step could also be automated using existing system identification or adaptive control techniques. It is important to note that this example is pedagogical and does not account for or attempt to mitigate disturbances. Nonetheless, this does not affect the generality of the result, as Embodied Visuomotor Representation solely determines the units of distance and remains compatible with any jumping algorithm.

3.5 Proofs

3.5.1 Uniqueness of solution of (3.10)

Equation (3.10) has a unique solution if and only if acceleration is non-zero for some time in the interval. (3.10) is reproduced below for clarity:

$$\min_{\frac{[d,x(t_0)]}{b}} \int_{t_0}^{t_0+T} \left(\Phi_W(t,t_0)\frac{d}{b} - \frac{x_1(t_0)}{b} - (t-t_0)\frac{x_2(t_0)}{b} - \int_{t_0}^{t}\int_{t_0}^{\sigma} u(\sigma_2)d\sigma_2 d\sigma \right)^2 dt$$

Proof The problem is a linear least squares problem. Thus it has a unique solution if and only if the signals $\Phi_W(t,t_0)$, 1, and $t-t_0$, are linearly independent over the interval $[t, t_0+T]$. Immediately, we can see that the signals 1 and $t-t_0$ are independent over any interval of non-

zero size. It remains to show under what conditions they are independent of $\Phi_W(t, t_0)$. Consider that, by definition, Φ_W and u are related by

$$\Phi_W(t, t_0)d = x_1(t) = x_1(t_0) - (t - t_0)x_2(t_0) - b\int_{t_0}^{t}\int_{t_0}^{\sigma} u(\sigma_2)d\sigma_2 d\sigma. \tag{3.26}$$

Thus, Φ_W is linearly independent from 1 and $t - t_0$ if and only if the double integral of u is independent from 1 and $t - t_0$. Suppose they are linearly dependent. Then there exists α_1, α_2 such that for all $t \in [t_0, t_0 + T]$

$$\alpha_1 + (t - t_0)\alpha_2 = \int_{t_0}^{t}\int_{t_0}^{\sigma} u(\sigma_2)d\sigma_2 d\sigma. \tag{3.27}$$

Taking the derivative of both sides twice implies $u(t) = 0$ for all t in the interval. Similarly, if $u(t) = 0$ for all t then the signals are linearly dependent. Thus, (3.10) has a unique solution, that is, Φ_W is linearly independent from 1 and $t - t_0$, if and only if $u(t) \neq 0$ for some time in the interval $[t_0, t_0 + T]$.

3.5.2 Uniqueness of solution of (3.11)

Equation (3.11) has a unique solution if and only if acceleration is non-zero for some time in the interval. It is reproduced below for clarity:

$$\min_{\frac{[b, x(t_0)]}{d}} \int_{t_0}^{t_0+T} \left(\Phi_W(t, t_0) - \frac{x_1(t_0)}{d} - (t - t_0)\frac{x_2(t_0)}{d} - \frac{b}{d}\int_{t_0}^{t}\int_{t_0}^{\sigma} u(\sigma_2)d\sigma_2 d\sigma\right)^2 dt.$$

Proof: The problem is a linear least squares problem and thus has a solution if only if the

double integral of u is linearly independent of 1 and $t - t_0$. The proof can proceeds identically to the previous section starting from equation (3.27).

3.5.3 (3.11) is an unbiased estimator

Proof: Equation (3.11) is a linear least squares problem. Thus, its solution is linear in Φ_W. In particular, the solution takes the form

$$\frac{[b, x(t_0)]^*}{d} = \langle J, J\rangle^{-1}\langle J, \Phi_W\rangle. \tag{3.28}$$

Here $J = [1, t-t_0, \iint u]$ is a vector of signals and $\langle\cdot,\cdot\rangle$ denotes the continuous time inner product evaluated over the interval $[t_0, t_0 + T]$. Consequently, if Φ_W is corrupted with zero mean noise $\mathcal{N}$, then expectation distributes and the estimate is unbiased. That is,

$$\begin{aligned} E\left[\frac{[b, x(t_0)]^*}{d}\right] &= E\left[\langle J, J\rangle^{-1}\langle J, \Phi_W + \mathcal{N}\rangle\right] \\ &= \langle J, J\rangle^{-1}\langle J, E\left[\Phi_W + \mathcal{N}\right]\rangle \\ &= \langle J, J\rangle^{-1}\langle J, \Phi_W + E\left[\mathcal{N}\right]\rangle \\ &= \langle J, J\rangle^{-1}\langle J, \Phi_W\rangle. \end{aligned} \tag{3.29}$$

Thus, the estimator is unbiased.

Further, the population average of $E[\mathcal{N}]$ will converge to zero as the length of the considered interval T increases. Thus, longer estimation intervals will result in better results. Additionally, the dependence of the solution on $\mathcal{N}$ can be decreased by increasing the magnitude of J and Φ_W. A system can accomplish this by making motions of greater amplitude.

3.5.4 The error of the estimate due to (3.11) decreases with the inverse of camera resolution

Suppose Φ_W is quantized due to its measurement by a camera of finite resolution. That is $\hat{\Phi}_W = \text{round}(\Phi_W/\Delta) * \Delta$ where $\Delta \propto 1/\text{res}$ and res is the resolution of the camera. Then the error in solving (3.11)'s deccreases with the inverse of the camera's resolution.

Proof: Recall from the previous section that

$$x^* = \langle J, J\rangle^{-1}\langle J, \Phi_W\rangle, \tag{3.30}$$

where we let x^* be the vector of solution parameters for notational convenience. By linearity of this solution, we have that the norm of the perturbed solution is

$$\|dx^*\| = \left\|\langle J, J\rangle^{-1}\langle J, \left(\Phi_W - \hat{\Phi}_W\right)\rangle\right\| \leq \left\|\langle J, J\rangle^{-1}\langle J, \cdot\rangle\right\| \|\Delta\| \tag{3.31}$$

Thus, we can expect errors in parameter estimates to decrease like $\Delta = 1/\text{res}$. This is confirmed by Figure 3.12. While Figure 3.12 is produced by solving Equation (3.14), the same argument holds for bounding the error in the estimates due to finite resolution.

3.5.5 Uniqueness of solution of (3.16)

The estimation of the parameters of linear systems is possible in general if and only if the input u is sufficiently exciting. This is a well known result that applies to all linear systems. An overview of the traditional approach is given in Chapter 2 of [67] which briefly develops the

general case and considers in detail the use of a single input to predict a single output using an FIR filter.

Here we develop a specific excitation condition for the case of (3.16). That is for the the case of multiple inputs and double integrated linear actuator dynamics. Further, because an impulse response is infinite-dimensional, we require that it be represented with a finite-dimensional basis. Using a finite dimensional basis is required for implementation on digital hardware. Further, while textbooks such as [67] focus on identifying finite impulse response (FIR) filters, this simply corresponds to the implicit assumption of a basis of delayed delta-diracs. Thus, identifying FIR filter parameters is a special case of the following result.

(3.16) is reproduced below for convenience:

$$\min_{\frac{[X(t_0),\dot{X}(t_0),G]}{d}} \int_{t_0}^{t_0+T} \left[\Phi(t,t_0) - \frac{X(t_0)}{d} - (t-t_0)\frac{\dot{X}(t_0)}{d} - \left(\frac{G}{d} * \int_{t_0}^{\cdot} \int_{t_0}^{\sigma} u(\sigma) d\sigma_2 d\sigma d\sigma \right)(t) \right]^2 dt,$$

Where G is a matrix value signal whose i,j'th entry is to be convolved with the j'th input to get its effect on the i'th output. Let the i,j'th entry be denoted by g^{ij}. Suppose each g^{ij} is parameterized by a set of basis functions. That is $g^{ij} = \sum_k c_k^{ij} f_k = {c^{ij}}^T f$ where f_k is a basis function and $f = [f_1, f_2, \ldots, f_n]$ is the vector of them.

The resulting finite-dimensional problem is:

$$\min_{\frac{[X(t_0),\dot{X}(t_0)]}{d},c} \int_{t_0}^{t_0+T} \left[\Phi(t,t_0) - \frac{X(t_0)}{d} - (t-t_0)\frac{\dot{X}(t_0)}{d} - \left(\underbrace{\left[{c^{ij}}^T f \right]}_{G/d} * \int_{t_0}^{t} \int_{t_0}^{\sigma} u(\sigma) d\sigma_2 d\sigma d\sigma \right)(t) \right]^2 dt, \tag{3.32}$$

In what follows, we show that the problem has a unique solution if and only if an excitation condition is satisfied. Additionally, it is necessary that the set of inputs $\{u_1, \ldots, u_M\}$ and basis functions $\{f_1, \ldots, f_N\}$ be respectively linearly independent.

Proof: The problem consists of three separate linear least squares problems, one for each element of Φ. It then immediately follows that these problems have a solution if and only if the sets of signals $S \coloneqq \{1, t - t_0\} \cup \{f_k * \int_{t_0}^{\cdot} \int_{t_0}^{\sigma} u_j(\sigma_2) d\sigma_2 d\sigma \mid \forall k, j\}$ are linearly independent. Note that this one condition is sufficient and necessary for all three problems.

Suppose that the set of signals S is linearly dependent. Then there exists scalars α_1, α_2 and a vector β such that for all $t \in [t_0, t_0 + T]$

$$\alpha_1 + (t - t_0)\alpha_2 = \sum_j \sum_k \beta_{jk} \left(f_k * \int_{t_0}^{\cdot} \int_{t_0}^{\sigma} u_j d\sigma_2 d\sigma \right)(t). \tag{3.33}$$

We can move the convolutions with f_k inside the double integral and take the derivative twice, revealing that:

$$\sum_j \sum_k \beta_{jk} \left(f_k * u_j \right) = 0 \tag{3.34}$$

Therefore, S is linearly independent if and only if $\exists t$ such that $\sum_j \sum_k \beta_{jk} \left(f_k * u_j \right)(t) \neq 0$. Consequently, it is necessary that the elements of f and u be linearly independent since

$$\sum_j \sum_k \beta_{jk} \left(f_k * u_j \right) = \sum_j \left(\sum_k \beta_{jk} f_k \right) * u_j = \sum_k f_k * \left(\sum_j \beta_{jk} u_j \right), \tag{3.35}$$

and so linear dependence of the elements of either u or f implies the existence of non-zero β such that the right hand side of (3.33) is zero. Then linear dependence of S holds with $\alpha_1, \alpha_2 = 0$.

Finally, we realize the excitation condition necessary for a unique solution. Define

$$\phi \coloneqq \mathrm{vec}\left\{\begin{bmatrix} (f_1 * u_1) & (f_2 * u_1) & \dots & (f_N * u_1) \\ (f_1 * u_2) & (f_2 * u_2) & \dots & (f_N * u_2) \\ \vdots & \vdots & \ddots & \vdots \\ (f_1 * u_M) & (f_2 * u_M) & \dots & (f_N * u_M) \end{bmatrix}\right\} = \mathrm{vec}\{u * f^T\}. \tag{3.36}$$

Where vec transforms a matrix to a vector and convolution is performed elementwise.

Then, if linear independence holds, $\forall \beta \neq 0$ we have that $\phi^T \beta \neq 0$. Squaring and integrating over time results in an excitation condition. That is, (3.32) has a unique solution if and only if the following excitation condition is satisfied:

$$\int_t^{t+T} \phi(\sigma)\phi^T(\sigma) d\sigma > 0. \tag{3.37}$$

In practice, it is well known that inputs that excite a wide range of frequencies result in good persistence of excitation [67].

3.6 Choice of safe contact speed v_s

Determining the safe contact speed during uncalibrated touching requires some assumptions by the designer or a higher level cognitive process within the robot. In general, and as with biological systems, there is no way to determine at what speed a system can make its first contact with the world without breaking. Below, we give two examples of how to set v_s for an embodied system.

3.6.1 Multiples of the touch target's size

Suppose the robot is to touch an object that is 0.5 meters wide, and the width of the touch target is the characteristic scale of vision d. Further, suppose the robot's designer wants the robot to come into contact with the target while traveling at 0.2 meters per second. Then, since the safe contact speed is equivalent to $0.2/0.5 = 0.4$ multiples of d per second, the safe contact speed should be set as $v_s = 0.4d$ within the robot's software. Subsequently, the robot can convert this setpoint to embodied units using its own estimate of d.

3.6.2 Multiples of maximum force

It is also possible to determine v_s by assuming a simple contact model, constant deceleration over some time, and requiring that the magnitude of the acceleration be equal to some multiple of the maximum force the robot can exert on itself. Let this multiple be α. Then, suppose the maximum acceleration achievable by the system's own actuators is u^{max}, and the expected deceleration period is $T > 0$. In that case, the average force experienced on contact is v_s/T, which should be less than αu^{max}. Thus, the safe speed can be chosen to satisfy $v_s <= \alpha T u^{max}$ where α and T can be intuitively chosen by the robot's designer.

Chapter 4: Extremum Seeking Controlled Wiggling for Tactile Insertion

Embodied Representation has so far been discussed from the perspective of measuring distances and estimating motor dynamics. However, manipulation is another context where Embodied Measurements, resulting from internal feedback, play a role. This is because what one must do to manipulate an object successfully is intrinsically tied to contact dynamics, which can be assumed to differ from object to object and so, the units of action also change from object to object. In what follows, a calibrated robot that uses the meter attempts to insert keys into locks. Access to these units does not help in determining how to insert the key further into the lock because of the unmodeled contact dynamics. The situation is resolved by using internal feedback as part of an extremum seeking control law. The result determines the direction to perturb the arm's average end effector pose in order to insert the key further into the lock without putting excessive strain on the key.

When humans perform complex insertion tasks such as pushing a cup into a cupboard, routing a cable, or putting a key in a lock, they wiggle the object and adapt the process through tactile feedback. A similar robotic approach has not been developed. We study an extremum seeking control law that wiggles end effector pose to maximize insertion depth while minimizing strain measured by a GelSight Mini sensor. Evaluation is conducted on four keys featuring complex geometry and five assembly tasks featuring basic geometry.

On keys, the algorithm achieves a 71% over 120 trials with 6-DOF perturbations, 84% over 240 trials with 1-DOF perturbations, and 75% over 40 trials initialized with vision. It significantly outperforms a baseline optimizer, CMA-ES, that replaces wiggling with random sampling. When tested on a state-of-the-art assembly benchmark featuring basic geometry, it achieves 98% over 50 vision-initialized trials. The benchmark's most similar baseline, which was trained on the objects, achieved 86%. These results, realized without contact modeling or learning, show that closed loop wiggling based on tactile feedback is a robust paradigm for robotic insertion.

4.1 Introduction

Imagine inserting a cup into a crowded cupboard. There are many objects in the way of where you want to put the cup, but somehow as the cup is inserted, they are pushed aside and the cup is placed in the appropriate location. A similar process occurs when routing a cable through a hole or inserting a key into a lock. Humans can accomplish all of them from a young age but robots still struggle. Further, general-purpose robots will need to solve such problems frequently, from peg-in-hole tasks featuring simple geometry, to more complex ones like key insertion. Key insertion offers unique challenges compared to the traditionally studied peg-in-hole because locks contain numerous internal contact surfaces which are difficult to model.

During such insertion tasks, humans often wiggle the object to be inserted while using tactile feedback from the fingers to adjust the process and maximize insertion depth. Thus, it is interesting to study generalizable insertion algorithms, inspired by human behavior, that use strain-like tactile feedback and control laws based on wiggling motions.

State-of-the-art robotic insertion methods typically focus on policies tailored to specific

Figure 4.1: By wiggling the 6 degree of freedom pose of an object grasped between two GelSight Mini tactile sensors and observing a strain-like quantity through the optical flow in the GelSight cameras, an extremum seeking control law performs insertion. All parameters are sinusoidally modulated simultaneously but at different frequencies, allowing for the estimation of a direction that minimizes strain and maximizes insertion depth along the Y axis. A supplementary video is available at `https://prg.cs.umd.edu/ESTac`.

objects [87,88] even when incorporating tactile feedback [87,89]. Approaches that aim for object generalization often rely on strong assumptions about initial conditions [90]. Recently, learning-based techniques have gained traction, but these are frequently trained and evaluated on the same set of objects [88,91]. In contrast, we demonstrate that real-time tactile feedback can successfully guide a learning-free wiggling-based insertion process that generalizes to new objects without requiring detailed contact-modeling, prior knowledge of 3D geometry, or strong assumptions on the starting pose.

We consider a compliant gripper, as shown in Figure 4.1, which grasps a key (in general, a

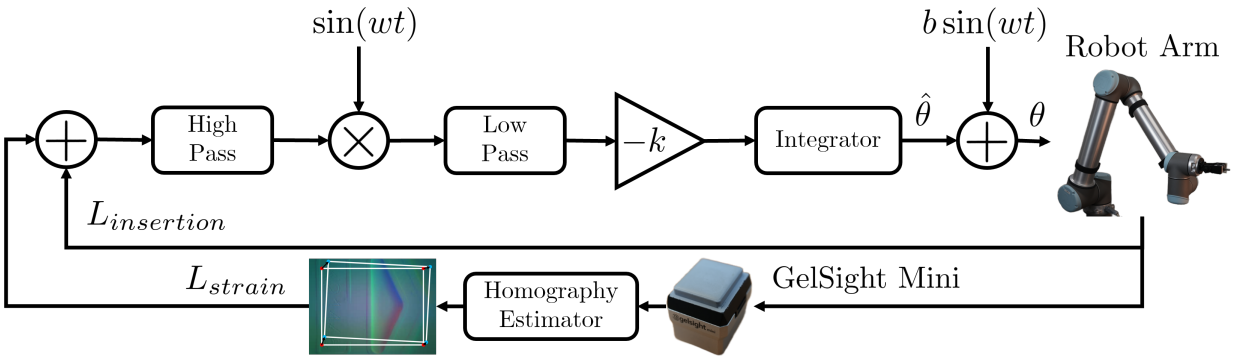

Figure 4.2: The extremum seeking controlled pipeline for wiggling-based tactile insertion. The instantaneous parameters θ control the pose of the tip of an object through a UR10 robot arm. The strain that the object exerts on the GelSight Mini's gel pad is observed via a displacement of the corners of a tracked patch in the sensor image feed, L_{strain}. The objective to be minimized is the sum of L_{strain} plus $L_{insertion}$, where $L_{insertion}$ represents the depth of insertion into the lock. The extremum seeking control seeks to minimize the objective by adjusting the parameter estimates $\hat{\theta}$. As is standard in Extremum Seeking Control, θ is a modulated version of $\hat{\theta}$ with each parameter modulated at a different frequency. The high pass filter removes the DC component from the objective signal, demodulation determines the slope of the objective's gradient, and the low pass filter averages the feedback signal with greater high-frequency attenuation than the integrator.

peg) between two GelSight Mini tactile sensors [92]. The pads experience strain when attempting to insert the key into a lock, which is observed through internal cameras that point toward the back of the gel. A control strategy based on wiggling is realized through Extremum Seeking Control, a 100-year-old method for online optimization of objective functions [93]. Due to its model-free formulation, it finds application in systems that are difficult to model, such as wind turbines and solar panels (which must handle changing weather), control of particle beams, and aircraft that experience flow instability [94]. Despite its generality, the method has received little attention in the robotics community.

When applied to tactile insertion, the extremum seeking control wiggles the 6 degree of freedom pose of the end-effector to estimate a descent direction that minimizes an objective depending on tactile feedback and end-effector pose. A block diagram is given in Figure 4.2. The method uses contact to correct pose, and so must be initialized near the opening, though not

further than existing methods when normalized by opening size. We validate this assumption by conducting trials initialized with vision.

A list of our contributions follows:

- A method for insertion, inspired by human behavior, that uses tactile-feedback to guide a wiggling process via Extremum Seeking Control that generalizes to all tested objects without per-object tuning.
- A method based on optical flow to measure strain-like forces on a GelSight Mini pad when grasping planar and rigid objects, such as keys.
- Detailed evaluation on key-in-lock insertion on four locks amounting to 360 trials while significantly outperforming a baseline based on CMA-ES [95] that randomly samples instead of wiggling.
- Vision-initialized experiments on key insertion and a robot assembly benchmark [88] (unseen until test time) that validate the method's assumptions and demonstrate remarkable performance on the assembly benchmark.

4.2 Related Work

Peg-in-hole problems are often considered in the field of robotics assembly. Common techniques have limitations when compared to our formulation based on closed loop wiggling including: requiring a contact model, the 3D model of the peg, or being restricted to specific object shapes or flat surfaces [87, 89, 96–103]. Others consider only small translation errors compared to the peg size [87, 102, 104], or prior knowledge of the hole position [101]. While

some works also use wiggling, they do so with feedforward motions, that is, without realtime adaptation based on feedback [91, 96, 97, 105, 106]. Often, methods focus on contact modeling for specific object and sensor configurations [87, 98]. Our method does not use a contact model.

Some works focus on problems adjacent to peg-in-hole including estimating the location of the grasped object within the grasp [107], vision based feedback [108, 109], the role of force or impedance feedback [89,91,99,106,110,111], learning based approaches [88,102,104,112–115], and the manipulator's mechanical design [116]. Our proposed approach is complementary to these ideas.

GelSight sensors were used as source of tactile feedback in several of these studies [87, 89, 100, 101, 104, 107]. One GelSight work also considers estimating a high-dimensional, dense optical flow on the gel pad surface during peg-in-hole insertion [117], in contrast our method is low dimensional and can track grasped object pose over long time periods.

Key insertion has received little attention. In [90], the problem of inserting a key into a bicycle lock is considered and solved using a bimanual approach while considering rotation uncertainty along a single axis and perfect translation estimates. In [118], the problem of localizing a keyhole is tackled using a pre-determined map of key-to-lock contact configurations. A few recent works have considered key-in-lock insertion as part of a suite of test objects [91, 106, 115, 119]. While [91, 106] incorporate a wiggling strategy, it is open-loop. Unlike these works, our approach considers closed-loop wiggling and shows that this paradigm can insert keys and objects, under 6D pose uncertainty, without sophisticated modeling, learning, or per-object tuning.

While Extremum Seeking Control has found extensive application in traditional engineering fields [94], to our knowledge, it has not been used for robotic insertion. The closest work is

an extremum seeking seeking control method for grasping with a gripper [120].

In summary, by developing a closed-loop wiggling process, we realize something unlike existing works. That is, a model-free approach to solve the tactile insertion task, making no specific assumptions about the object's shape, contact model, or restricting the uncertainty of the holes position to particular degrees of freedom.

4.3 Methods

Our algorithm that realizes closed-loop wiggling based on tactile feedback has three parts. First, we describe the tactile feedback using strain measurements taken by a GelSight Mini tactile sensor and computed using a Lucas-Kanade style homography tracker. Next, we detail the objective function minimized by the extremum seeking control. Finally, we describe the extremum seeking control law, which sinusoidally perturbs the 6 degree of freedom pose of the end-effector to estimate a descent direction and minimize the objective function. A block diagram is given in Figure 4.2.

4.3.1 Tactile Strain Measurement

The gripper configuration is shown in Figure 4.1. Two GelSight Mini tactile sensors are pressed against the left and right side of the key head. One tactile sensor is off and functions as a soft barrier. The second sensor is on, and the 2D imagery captured from the internal camera is exemplified in Figure 4.3. Because the contact pads are compliant, when pressure is applied to the tip of the key, the key moves with respect to the gel pads and warps the 2D imagery. Further, because the head of key is flat, this warp is described as an 8-parameter homography. Thus, a

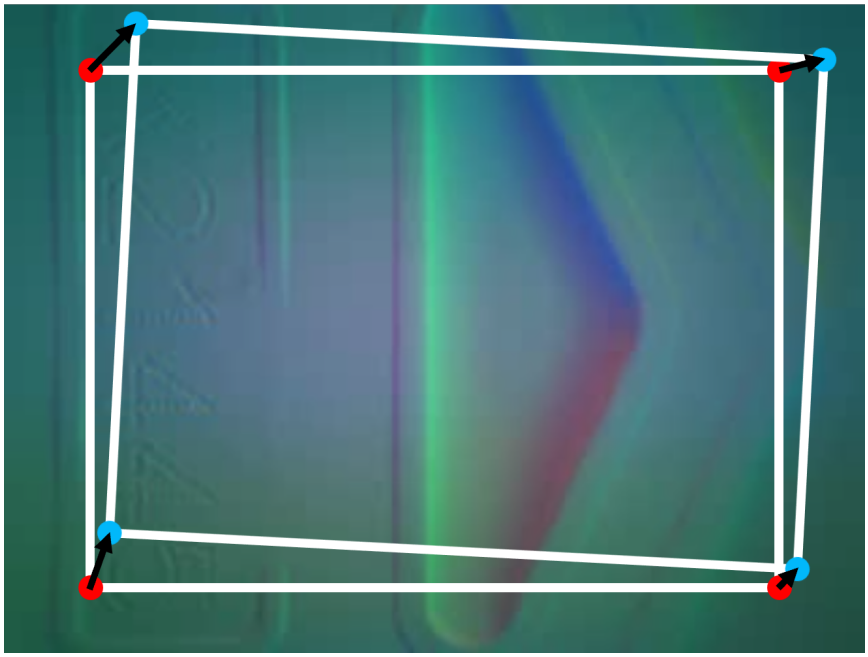

Figure 4.3: A strain-like measurement is measured directly from the images returned by the GelSight Mini sensor. Each incoming frame is iteratively registered with a Lucas-Kanade style homography estimator to the first frame. The tracked patch has 10% margins with respect to the full frame and is exemplified by the area within the white box with red corners. The Euclidian norm of the corner displacements in pixels from their original location is used as the strain-like quantity L_{strain}.

Lucas-Kanade style homography estimator is used to continuously warp back the current image to the first frame. The optimization is warm-started at each iteration using the previously estimated homography. The implementation is based on [42] and uses the 4 corner parameterization suggested in [121]. Before passing through the tracker, frames are resized to 320×240 resolution. The tracked patch is also initialized to the entire frame with 10% margins, as illustrated in Figure 4.3.

The displacement of the four corners of the tracked patch from is used as the strain metric. That is

$$L_{strain}(t) = \sqrt{\sum_{i=1}^{4} \|p_i(t) - p_i(t_0)\|^2}, \tag{4.1}$$

where $p_i(t)$ is the pixel location of the i'th corner at time t as estimated by the homography tracker and t_0 is the time of the first frame. To avoid the effects of noise when pressure is low $L_{strain} < 3$ pixels is reported as 0. The strain feedback is computed at 10-16 Hz. The update rate is limited by a combination of OpenCV's Python V4L2 camera interface, USB 2.0 bandwidth limits, and GelSight Mini's hardware requirement that images be retrieved at the high resolution of 3280×2464 in the Motion-JPEG format.

4.3.2 Control Objective

To encourage the extremum seeking control to guide the key into the lock, an additional loss term $L_{insertion}$ is defined

$$L_{insertion} = |Y - (Y(t_0) - d)|, \tag{4.2}$$

where Y is the position of the tip of the key in meters along the axis to be inserted into the lock and d is the depth of the keyhole. If $L_{insertion} < 0.5$ millimeters, the algorithm is stopped and insertion is considered a success.

The final objective function is then

$$L = L_{insertion} + \lambda L_{strain}, \tag{4.3}$$

where $\lambda = 0.0005$.

4.3.3 Extremum Seeking Control

Classical extremum seeking control as described in [94] is applied directly. A block diagram is given in Figure 4.2. The estimated parameters $\hat{\theta}$ are the 6 degree-of-freedom pose of the tip of the key. That is $\hat{\theta} = [X, Y, Z, \alpha, \beta, \gamma]^T$ where α, β, γ are $x - y - z$ intrinsic Euler angles corresponding to the orientation of the tip of the key.

As is standard in extremum seeking control, the applied parameters $\theta(t)$ are modulated according to

$$\theta(t) = \hat{\theta}(t) + \text{diag}\{b\} \sin(wt), \tag{4.4}$$

Where b and w are vectors with the same dimensionality as θ, $\text{diag}\{\}$ is the operator that constructs a diagonal matrix from a vector, and $\sin$ is applied element-wise. In all the experiments, $b_{1,2,3} = [0.2, 0.2, 0.5]^T$ millimeters, $b_{4,5,6} = [0.675, 0.675, 0.675]^T$ degrees and $w = [0.9, 0.83, 0.7, 1.05, 1.0, 0.95]^T$ Hz. These perturbation parameters were chosen by hand, without much difficulty. It is likely that more performant parameters exist.

Following the extremum seeking control pipeline, illustrated by Figure 4.2, the objective L is sampled at each θ, resulting in a signal $L(t)$. $L(t)$ is high pass filtered with a first-order high pass filter with a 0.7 Hz cutoff, demodulated with $\sin(wt)$, and low pass filtered with a first-order filter with a cutoff of 1.59 Hz. This final signal is the feedback error multiplied by a control gain k to determine $\dot{\hat{\theta}}$. Precisely,

$$\dot{\hat{\theta}}(t) = -\text{diag}\{k\} \left[g_{lpf} * x_{demod} \left[g_{hpf} * L \right] \right](t). \tag{4.5}$$

Here $x_{demod}(t) = \text{diag}\{\sin(wt)\}$ is the demodulating signal, g_{lpf} and g_{hpf} are the impulse re-

sponses of the low and high pass filters respectively, $*$ denotes convolution, $k = [0.7, 1.1, 0.7, 10.0, 10.0, 10.0]^T$. A UR10 robot arm tracked θ using linear servoing.

4.4 Experiments and Results

Three sets of experiments were performed involving four distinct lock types and five objects from the AutoMate assembly benchmark [88], representing complex and basic geometry respectively. The first experiment involved 240 trials, and the second 120, and the third 90, for a total of 450 trials. The first experiment tested perturbing a single degree of freedom (DOF) from an initial pose aligned with the lock entrance. The results demonstrate which types of perturbations the method is sensitive to. The second experiment tests initialization from randomly sampled 6-DOF poses, thus testing the methods robustness to significant misalignment. The third experiment tests the assumption of initial alignment with the hole, up to a tolerance, by using a camera to estimate the pose of the insertion target and subsequently set the initial position. Figure 4.4 shows the objective and pose during a key in lock insertion starting from a random 6-DOF pose.

We establish a baseline for the extremum seeking control law by comparing it to CMA-ES, a gradient free optimization algorithm that is widely used. Instead of wiggling, it randomly samples. Additionally, in the vision initialized trials, we test on the same five objects used in AutoMate's [88] vision-initialized trials. AutoMate's baselines are trained on the objects tested on. To our method the objects were unseen.

Other approaches based on RL were not replicated due to missing code [89, 90, 104, 113–115, 122], unfeasible assumptions on the objects [87, 98, 100, 103, 105], starting error limited

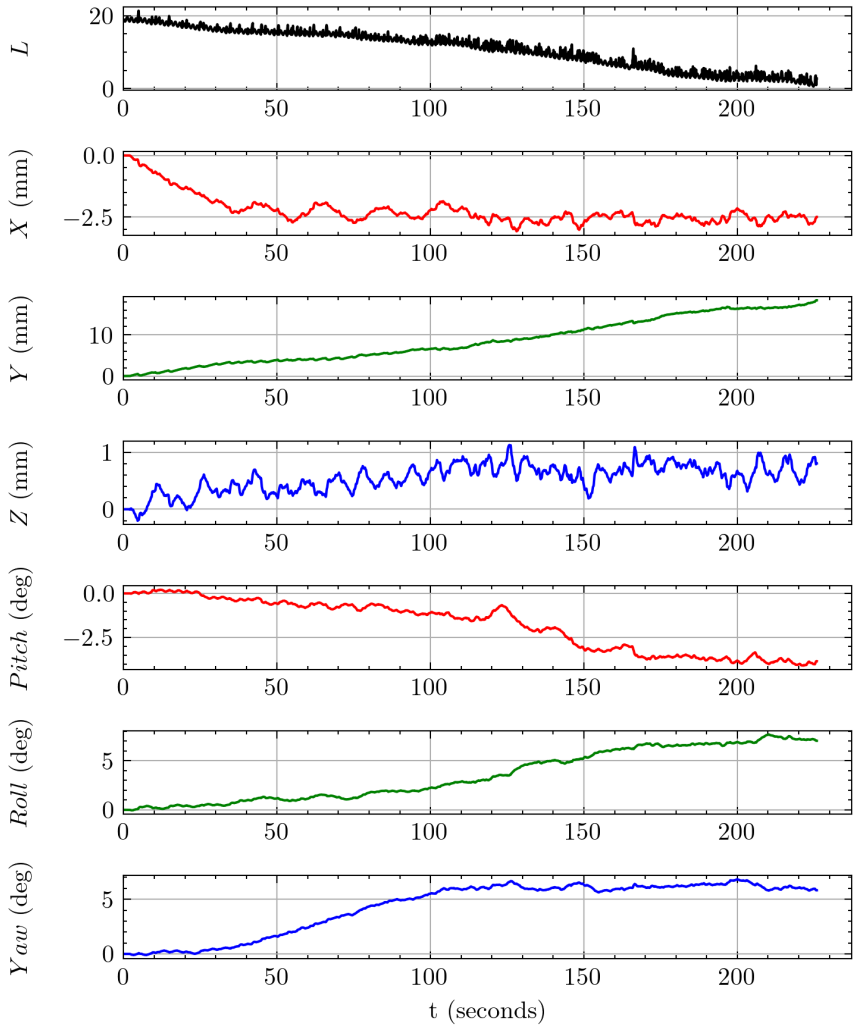

Figure 4.4: The loss and estimated parameters (end-effector pose) converge as the key is inserted. The Y parameter increases steadily because it corresponds to the insertion axis. The trial pictured was initialized with $[1.1, 0.0]$ millimeters of translation along the X, Z axis and $[3.4, -7.4, 5.7]$ degrees of rotation about the X, Y, Z axis.

Figure 4.5: The four types of key and lock pairs tested. L1 is a common pin-tumbler lock used on front doors. L2 is a dimpled cam lock that uses pin-tumblers like L1, but they also press on the sides of the key into the dimples. L3 is a tubular lock with a circular shape that must be pressed into the lock's circular opening. L4 is a disc-detainer which features rotating discs inside the lock that must be aligned.

to certain axis [90], prior knowledge of contact model [96], perfect knowledge of desired final pose [91], prior knowledge of the object model during insertion [89], or incompatible sensors and contemporaneity [119].

4.4.1 Lock Types

The four locks tested are illustrated in Figure 4.5. All feature complex geometry. The first lock is a cylindrical pin-tumbler lock. The ridges on the top of the key actuate the pin tumblers inside the lock. The second is a dimpled key cam lock that uses dimples on the sides of the

Lock	Translation Misalignment (mm)								Rotation Misalignment (deg)												CMA Rotation Misalignment (deg)							
	X				Z				X				Y				Z				X				Z			
	-2.5	-1.9	1.9	2.5	-2.5	-1.9	1.9	2.5	-10	-5	5	10	-10	-5	5	10	-10	-5	5	10	-5	-2	2	5	-5	-2	2	5
Success Rate (%)																												
L1	100	100	100	100	100	100	100	0	100	100	100	100	100	100	100	33	100	100	100	100	0	100	67	0	0	33	0	0
L2	67	67	100	0	33	100	33	67	100	100	100	67	100	100	100	100	67	100	67	67	0	0	33	0	0	0	33	0
L3	100	100	100	0	0	100	100	0	100	67	100	100	0	100	33	0	100	33	67	100	33	100	100	67	100	67	67	67
L4	100	100	100	100	100	100	100	100	100	100	100	67	100	100	100	100	100	100	100	100	67	100	100	33	67	0	100	0
Insertion Time (sec)																												
L1	322	96	93	117	215	106	93	-	232	116	113	217	505	91	85	120	160	123	136	181	-	249	265	-	-	231	-	-
L2	335	195	164	-	256	196	739	338	188	153	153	185	201	308	147	203	300	164	170	490	-	-	213	-	-	-	267	-
L3	335	89	83	-	-	118	53	-	145	36	80	155	-	44	62	-	85	41	52	106	73	48	65	56	61	55	47	112
L4	86	90	97	88	104	95	106	86	100	100	104	99	111	91	90	68	115	113	108	96	142	162	208	185	220	-	194	-

Table 4.1: Success rate and insertion time versus single parameter perturbations of initial pose.

key to actuate pin-tumblers inside the lock. Compared to a standard pin-tumbler, dimpled keys require some variation in force to insert due to the way the pin-tumblers press into the dimples. The third lock is a tubular cam lock. As shown in Figure 4.5 the tubular key is circular instead of rectangular and thus has a significantly different form than the pin-tumbler and dimpled keys. Finally, the fourth lock is a padlock disk-detainer. Disc-detainer locks use rotating disks that must be aligned. The locks also have differently shaped front faces. L1 and L4's have front faces that generally slope toward the opening. However, L2 and L3 feature regions near the opening that slope away, or are flat.

4.4.2 Initial Pose Perturbed on One Axis

The first experiment demonstrates the robustness to perturbations along one translation or rotational axis of the initial key pose away from alignment with the lock's keyhole. For each lock, translation perturbations of -2.5, -1.9, 1.9, and 2.5 millimeters were tested along the plane parallel to the lock surface. Additionally, initial rotation perturbations of -10, -5, 5, and 10 degrees were applied to each orientation axis individually. A 0-millimeter and 0-degree setting corresponds to

a near-perfect alignment between the key and the lock. The translation misalignment considered is larger than the hole size. In contrast, state-of-the-art works that, like ours, do not incorporate a search strategy [87, 97], assume smaller translations relative to hole size. Three tests were performed per initial pose. The results are given in Table 4.1.

The success rate over the smaller perturbations (-1.9 and 1.9 mm in translation) and (-5 and 5 degrees in rotation) was 90.8% over 120 trials with a mean insertion time of 122 seconds. The success rate over the large perturbations (-2.5 and 2.5 mm in translation) and (-10 to 10 degrees in rotation) dropped to 76.7% over 120 trials with a mean insertion time of 171 seconds. The overall success rate over the 240 trials was 83.8% with a mean insertion time of 147 seconds.

Lock L4 (Disc-Detainer) was the easiest as only one trial failed, resulting in a 98% success rate overall with mean insertion times of 98 seconds. Lock L1, L2, and L3 achieved 93%, 77%, and 65% overall with mean insertion times of 169, 232, and 105 seconds respectively. The drop in performance on lock L2 is expected because the lock surface mainly slopes away from the keyhole, which makes it difficult for the extremum seeking control to enter. Similarly, lock L3 is flat on the surface of the keyhole and so it proved difficult for the extremum seeking control to find the hole, especially when the perturbations were large.

As a baseline, we conducted experiments using CMA-ES [95] using smaller angular perturbations of -5, -2, 2, and 5 degrees along the X and Z axes. The overall success rate was 42% over 96 trials. Detailed results are reported on the right side of Table 4.1. CMA-ES fails with small 2-degree perturbations, whereas extremum seeking performs well at 5 and even 10-degree perturbations. CMA-ES was not tested with translation perturbations because the performance on rotation was convincingly poor. CMA-ES had to be modified to avoid breaking the GelSight. Sigma adaptation was disabled as it excessively increased the pose perturbations. Even then, the

Type of Lock	Translation Misalignment (mm)								Rotation Misalignment (deg)											
	X				Z				X				Y				Z			
no. of trials	(-2.5,-1.9) 2	(-1.9,0) 9	(0,1.9) 15	(1.9,2.5) 4	(-2.5,-1.9) 2	(-1.9,0) 10	(0,1.9) 13	(1.9,2.5) 5	(-10,-5) 8	(-5,0) 8	(0,5) 9	(5,10) 5	(-10,-5) 6	(-5,0) 6	(0,5) 11	(5,10) 7	(-10,-5) 10	(-5,0) 7	(0,5) 6	(5,10) 7
Success Rate (%)																				
L1	100	67	73	100	0	90	85	60	100	62	78	60	67	67	91	71	100	100	67	29
L2	50	67	80	0	100	80	54	40	62	62	78	40	67	67	64	57	70	43	67	71
L3	50	56	47	0	0	50	46	40	62	12	33	80	33	67	36	43	70	14	67	14
L4	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
Insertion Time (sec)																				
L1	125	277	204	169	-	226	177	285	205	261	173	225	173	161	270	449	151	161	167	383
L2	1360	387	429	-	545	416	385	860	389	430	594	289	303	571	507	594	346	556	410	587
L3	210	301	406	-	-	325	341	445	277	342	403	406	406	250	324	170	487	320	262	419
L4	135	130	138	125	132	107	139	173	114	109	123	223	165	103	150	106	106	146	129	153

Table 4.2: Marginal distributions of success and insertion time for Extremum Seeking Control with 30 random initial poses tested on all four locks. Our method achieves an average insertion time of 262 seconds with a 71% success rate. The baseline, CMA-ES, failed on 5 consecutive trials on L1, L2, and L4 and succeeded on 2 out of 5 trials on L3.

	L1	L2	L3	L4	00340	00320	00346	00015	00296
Pitch (deg)	0.504 ± 0.415	0.522 ± 0.463	2.174 ± 0.710	0.636 ± 0.521	0.882 ± 0.545	4.677 ± 2.174	4.677 ± 2.174	3.064 ± 1.971	1.768 ± 0.913
Roll (deg)	2.490 ± 0.681	1.337 ± 0.500	0.879 ± 0.325	0.511 ± 0.319	96.289 ± 0.470	2.640 ± 0.646	2.640 ± 0.646	62.755 ± 35.238	89.953 ± 0.922
Yaw (deg)	2.298 ± 0.577	1.441 ± 0.850	0.696 ± 0.555	0.922 ± 0.631	0.849 ± 0.660	3.319 ± 2.169	3.319 ± 2.169	1.984 ± 1.293	2.547 ± 0.913
X (m)	0.002 ± 0.001	0.001 ± 0.001	0.001 ± 0.001	0.001 ± 0.001	0.002 ± 0.001	0.002 ± 0.002	0.002 ± 0.002	0.002 ± 0.001	0.001 ± 0.001
Y (m)	0.002 ± 0.001	0.003 ± 0.001	0.003 ± 0.001	0.003 ± 0.001	0.002 ± 0.001	0.002 ± 0.001	0.002 ± 0.001	0.003 ± 0.002	0.002 ± 0.001
Z (m)	0.002 ± 0.001	0.003 ± 0.002	0.005 ± 0.002	0.003 ± 0.002	0.004 ± 0.002	0.007 ± 0.002	0.007 ± 0.002	0.001 ± 0.001	0.004 ± 0.002

Table 4.3: Vision based initialization's rotation and translation error (mean ± std) estimated from 100 samples per object. Objects with high roll error are symmetric about the roll axis.

movements to the requested poses were frequently aborted because the measured strain exceeded a safety threshold.

4.4.3 Initial Pose Determined Randomly

Robustness to initial pose was evaluated using 30 random samples of end-effector pose. Translations parallel to the lock surface were drawn from a uniform distribution over -2.5 to 2.5 mm, and rotations in Euler angles were drawn from a uniform distribution over -10 to 10 degrees. The distributions of the success rates are given in Table 4.2.

Our method's average success rate over the uniformly sampled smaller translation perturbations (-1.9 to 1.9 mm) was 74% in X and 75% in Z with mean insertion times of 269 and 240 seconds respectively. Over the smaller rotational perturbations (-5 to 5 degrees), the expected success rate was 66% in X, 73% in Y (insertion direction), and 69% in Z with mean insertion times of 274, 241, and 253 seconds respectively. The success rate over larger translation perturbations (-2.5 to 2.5 mm) was 58% in X and 57% in Z with mean insertion times of 233 and 355 seconds respectively. Over the larger rotational perturbations (-10 to 10 degrees), the expected success rate was 77% in X, 67% in Y (insertion direction), and 72% in Z with mean insertion times of 247, 291, and 268 seconds respectively. The overall success rate was 71% over 120 trials with a mean insertion time of 262 seconds. Lock L4 (Disc-Detainer) was the easiest, with zero failures and a mean insertion time of 134 seconds. Lock L1, L2, and L3 achieved 77%, 63%, and 44% overall with mean insertion times of 210, 465, 351 seconds respectively. As a baseline, we tested CMA-ES with 5 random poses on each lock and succeeded only twice, both times on L3.

4.4.4 Vision Based Initialization

The pose uncertainties assumed by the previous experiments were validated with two experiments using vision based initialization. Similar to AutoMate, a hand mounted camera was used to estimate the object's pose using [123] and [124]. Experiments were conducted using the 4 locks and also 5 assemblies from a robot assembly benchmark [88]. The assemblies are pictured in Figure 4.6. One hundred random poses sampled from a cube 20 cm on each side, with the closest face 20 cm from the opening. Orientation was sampled uniformly from $SO(3)$ and rejected if the plug was outside the field-of-view or the roll was greater than 45 degrees. The resulting yaw,

pitch, and roll fell between approximately ±40, ±25, and ±45 degrees respectively. The pose was then estimated from vision and the resulting translation and orientation error and standard deviation for each object are listed in Table 4.3. The errors are typically within a few millimeters or degrees, which is well within the margins tolerated by our method on locks.

Additionally, 10 vision initialized insertion trials were conducted on each object (90 trials total). The starting pose was randomly sampled as when estimating pose uncertainty. The success rates are reported in Table 4.4. The average success rate for the locks and assemblies was 75% and 98% respectively. The performance on locks was comparable to the average performance in the previous experiments, validating our assumptions. The average performance on AutoMate's assemblies was higher than AutoMate's vision-initialized generalist and specialist baseline's which achieved 86% and 90% respectively. It should be noted that AutoMate's vision-initialized trials also estimated the pose of the object in hand. However, our average vision-initialized performance is higher than AutoMate's highest manually initialized basline performance, which was 96%. These results clearly show that our method generalizes well to other types of insertion, significantly outperforming the state-of-the-art approach without per-object training, or in general, learning.

AutoMate's baseline was evaluated on assemblies with 1 millimeter, 45 degree chamfers on contacting edges. The released dataset is not chamfered. For our evaluation, we applied smaller 0.707 millimeter chamfers. We also evaluated on non-chamfered assemblies, resulting in a substantially lower average score of 50% (Ours-NC in Table 4.4). These results indicate our method benefits from a small amount of chamfering. It is unknown how AutoMate performs on non-chamfered assemblies. While AutoMate's training code is available, the code for deploying on a real robot is not.

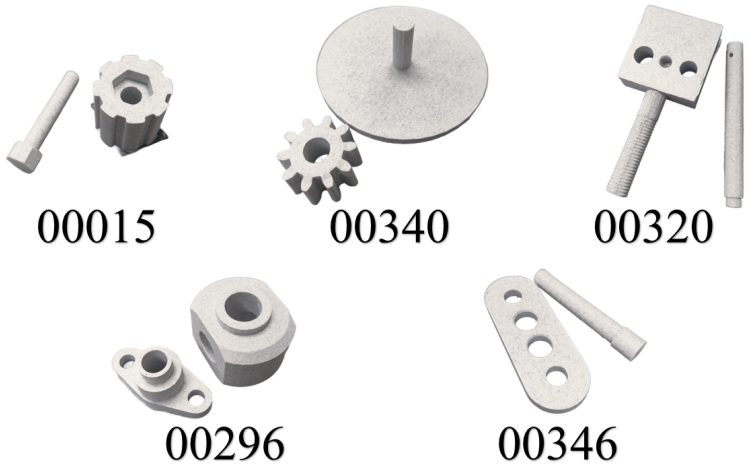

Figure 4.6: We evaluate vision initialized insertion on 5 objects with basic geometry from the robot assembly benchmark AutoMate.

Lock	**Ours**	**Assembly**	**AutoMate [88]**	**Ours**	**Ours-NC**
L1	80	00340	80	100	30
L2	80	00320	80	100	80
L3	50	00346	80	100	40
L4	90	00015	100	100	60
		00296	90	90	40
Mean	**75**	**Mean**	**86**	**98**	**50**

Table 4.4: Vision initialized success rate (%) on locks and assemblies (10 trials each). AutoMate's baseline is trained on the assemblies. AutoMate evaluated on chamfered assemblies, but the released dataset is not chamfered. Ours-NC denotes our method on non-chamfered assemblies.

4.5 Discussion and Future Work

The value of a closed loop wiggling process guided by tactile feedback in complex insertion problems has been demonstrated using an extremum seeking controller. A single set of a parameters generalizes to multiple locks, which feature complex geometry, and achieves remarkable success on a robot assembly benchmark which features basic geometry.

The method significantly outperforms a baseline based on CMA-ES which struggles to correct the orientation of the key so that it can enter the keyhole. The difference in performance

can be explained by considering the major difference in sampling strategy between the two algorithms. Extremum seeking continuously moves the key into different poses with a wiggling like motion while simultaneously sampling the sensor. This allows it to press the key into the lock while also adjusting orientation. In contrast, CMA-ES samples random poses which results in a much slower sampling rate and far fewer orientation samples while pressing into the lock.

The performance on the assembly benchmark was significantly higher than that of the learning-based approach, validating the importance of closed-loop wiggling. Moreover, these experiments validate the effectiveness of the proposed approach across a broader range of geometries and insertion types. For instance, objects 00340 and 00296 represent cases of "inverse" insertion, where the socket is held in hand instead of the plug. Finally, the results reinforce the assumption that key insertion constitutes a particularly challenging task.

Next, we analyze the primary reasons for failure on each lock. Lock L1 (pin-tumbler) has sharp edges along the top and bottom of the key hole that the key can become stuck in. In this case, our method is too naive to detect that the key is stuck, or wedged, and does not back away (12 failure cases). Lock L2 (dimpled) has sharp edges on the key that sometimes became caught on the entrance of the lock (7 failure cases). Additionally, L2 has a raised keyhole as can be seen in Figure 4.5. Thus, the extremum seeking control law sometimes moved the key over the edge and got stuck (11 failure cases). Finally, lock L3 (tubular) achieved the lowest success rate because orientation about the Y axis (insertion axis) is much more important than in L1, L2, or L4. Additionally, the surface of the lock is flat, and thus provides little to no information for the extremum seeking control law to use to proceed into the lock (33 failure cases). These two issues also occasionally caused a maximum limit on strain to be exceeded (3 failure cases). The reasons for five cases of failure were not recorded.

Most failure modes are due to the direct application of extremum seeking control. While extremum seeking can use its sinusoidal perturbations to escape small local minima and saddle points, such as those caused by the the teeth of a key, it cannot always escape. Regardless, the performance on the key and lock problem, which features complex geometry, is striking. There are many ways to improve. In particular, approaches that incorporate learning in a manner similar to [88] and [91] are promising. Additionally, it is of interest to learn task specific wiggling patterns and consider more specific tactile feedback than the proposed displacement based strain analogue. Further, we note that the primary technical assumption of our implementation, grasping planar and rigid local surfaces whose displacement can be represented as a homography, could be removed by replacing the tactile strain estimator, without altering the overall approach. Finally, the current work relies on a tight grasp; future work can consider adjusting grasp strength and compensating for slippage.

The current approach also admits several immediate paths towards improving the mean insertion time and success rate. In particular, the control parameters were tuned only to demonstrate basic efficacy Additionally, the objective function is currently linear in each argument and which results in the slow, linear convergence seen in Figure 4.4.

4.6 Conclusion

Closed-loop wiggling using tactile feedback has significant potential in robotic insertion problems involving complex geometries. Future work can consider many adaptions. In particular, we propose several insertion problems in Table 4.5 which feature complex geometries that humans often tackle using wiggling. We categorize them as “direct”, “through”, and and “crowded”

Type	Examples	Tactile Modalities
Direct	Key, Seatbelt	Grasp strain
Through	Cable routing Putting on belt	Grasp strain Bimanual contact
Crowded	Putting a cup in a cupboard Packing a box	Grasp strain Manipulator contact

Table 4.5: Tasks completed by humans with wiggling that are promising directions for future work.

and propose the tactile modality that is likely to be most important during the wiggling process.

Chapter 5: Artificial Microsaccade Compensation: Stable Vision for an Ornithopter

Animals with foveated vision, such as humans, experience microsaccades, small, rapid eye movements that they are not aware of. Inspired by this phenomenon, we describe a method for "Artificial Microsaccade Compensation" which can stabilize video captured by a tailless ornithopter. The platform has resisted attempts to use camera-based sensing due to aggressive shaking caused by flapping at 12-20 Hz. Our approach directly minimizes changes in image intensity by optimizing over 3D rotation represented in $SO(3)$. The method can render a stabilized video in real time, which is suitable for human viewing and free from apparent distortion. Further, when adapted to fixation followed by saccades, as is appropriate for robot vision, it can dramatically reduce inter-frame motion while also benefiting from an efficient recursive update. When compared to Adobe Premier Pro's warp stabilizer, which is widely regarded as the best commercial video stabilization software available, our method achieves significantly higher quality results while also running in real time. Videos are available at `https://prg.cs.umd.edu/AMS`.

5.1 Introduction

Flapping flying robots, or ornithopters, stand to revolutionize micro air vehicles, because they are safer, quieter, resilient to contact, and more aesthetically appealing than their rotor based

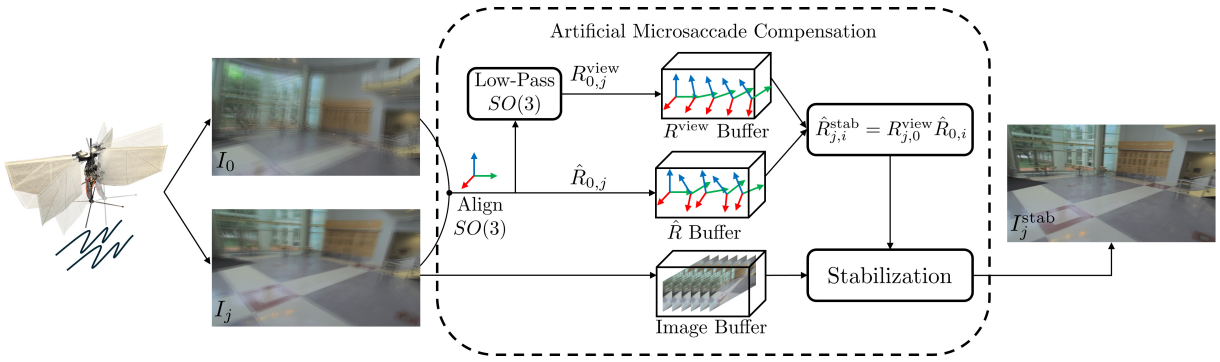

Figure 5.1: Artificial Microsaccade Compensation allows stabilizing unstable videos, such as those captured by the Flapper Nimble+, by directly matching incoming frames to a periodically updated template frame. Under the assumption of small rotational disturbances, a direct optimization of the mean squared error between images is used to continuously update an orientation estimate $\hat{R}$. Subsequently, a smoothed viewpoint R^{view} is computed. The simplest option for the stable viewpoint is a constant orientation. However, a more sophisticated approach that smoothly tracks the system's viewpoint can be realized with a low-pass filter on the group of rotation matrices $SO(3)$. The frames and rotations are buffered and used in the Artificial Microsaccade Compensation process, which combines multiple frames, taken at multiple times, to realize a high quality and stable video that is free from distortion.

counterparts. However, their wing-based actuation induces shaking that makes camera-based perception difficult [125]. In particular, tailless flapping robots are of interest because they can hover and are more agile than their tailed counterparts [126–129] and can even imitate agile insect behaviors [130]. However, tailless ornithopters also shake aggressively, which creates difficulties when using vision-based sensing [131–133]. The deteriorated visual sensing is a major obstacle on the road to real-world application of flapping wing robots, as it makes it hard to use cameras for autonomous flight and applications.

An interesting parallel can be drawn to animals with foveated vision, such as humans, who constantly experience rapid, small eye movements called microsaccades. These small rotational eye movements are not perceived in our consciousness, and while their purpose or lack thereof is still a matter of debate, there is a consensus that the brain employs several techniques to mask, or suppress, these movements so that they do not directly influence our perception of the world as

being viewed from a stable orientation [134]. Thus, it is of interest to develop an artificial method for compensating microsaccade like motion for robots that shake.

In this paper, we describe a real-time method for video stabilization, which we call "Artificial Microsaccade Compensation". It directly minimizes differences in image intensity, without relying on feature matching. The resulting rotations are used to estimate a stable, slowly varying orientation to which frames are rendered via a stabilizing rotation. The optimization is conducted over the rotation group of $SO(3)$ using an efficient, equivariant approach based on the inverse compositional Lucas-Kanade method [42].

While direct methods for computing image motion have been known for some time [135], modern open-source libraries lack the capabilities required to directly estimate nonlinear warps, such as 3D rotations, in a direct and efficient manner without resorting to feature matching. Consequently, we open-source our approach in an "Artificial Microsaccade Compensation" package, in the hope that other practitioners with shaking robots can stabilize their image streams by rendering them to a stable viewpoint.

We make the following contributions:

- A framework and implementation for "Artificial Microsaccade Compensation". That makes clear how a robot can represent a video from a stabilized orientation without apparent distortion, and which can be adapted for other applications.

- An efficient implementation of a direct Lucas-Kanade style method for image rotation estimation based on equivariance and optimization in $SO(3)$ with updates computed in the Lie Algebra $so(3)$. To the best of our knowledge, an equivalent implementation is not available.

- Evaluation of the method in a particularly difficult stabilization problem arising from the shaking of a tailless ornithopter, the Flapper Nimble+, which has thus far admitted only limited application of camera based sensing [136]. The result is the first stable videos resulting from a camera on the platform.

5.2 Related Work

Video stabilization is a well studied topic due to its numerous applications. Several distinct techniques have evolved [137–139]. Widely known techniques attempt to estimate the scene geometry and the camera's motion, then re-render the video to look as if it came from a camera that smoothly rotates and translates [140]. Another popular technique is subspace methods, which split the image into many distinct parts that are stabilized separately, with some constraints on smoothness [141]. Many methods result in gaps between separately stabilized regions of the image, which has led to the study of inpainting techniques that automatically fill those gaps [142, 143]. The predominant techniques rely heavily on feature matching, which requires significant effort to ensure feature trajectories are long enough and smooth enough to achieve satisfactory results [144, 145]. More recently, methods that incorporate the aforementioned concepts, but are based on deep learning and advanced representations such as Gaussian Splatting, have achieved impressive results but are typically far from achieving real-time performance and require significant training data [139, 146].

Robotic applications require real-time, robust performance, and it is well known that feature matching across rapidly moving frames can fail because of motion blur. Further, our emphasis on microsaccade like motions means any estimated image motion should be constrained

to that of a 3D rotation, which is difficult to enforce on feature trajectories. Stabilization based on 3D rotation and known intrinsics has been previously studied, but feature matching is still assumed [147]. Optical image stabilization based on angular velocity sensors have also been studied for some time [148]. However, applications typically focus on hand stabilization, which is relatively low bandwidth (<10 Hz) [149]. Deep learning based variants of angular velocity sensor based stabilization have also appeared [150]. In contrast, our desired application of stabilizing the video from a tailless ornithopter involves a base frequency of 12-20 Hz, which is augmented with higher order harmonics. Additionally, we find that the IMU on our tailless ornithopter does not accurately estimate the system's angular velocity during flight, at least as it comes configured by the manufacturer. Lastly, using gyroscopic sensors introduces additional complications, such as time synchronization and calibration of the gyroscopic sensor, which are often not well handled by low-cost hardware.

Flapping flying machines, or ornithopters have been studied in one way or another since the ancient greeks [133]. A particularly popular platform is the Flapper Nimble+, which is a tailless ornithopter whose original design was used to demonstrate that fruit flies rely on torque coupling to perform rapid bank maneuvers [130]. As a result, the platform is quite agile, however, it also shakes aggressively because it cannot rely on the tail for damping. Moreover, the rotations due to shaking are irregular since the two wing pairs are actuated independently to generate the moments necessary for flight maneuvers. There is potential for significant advancements in the mechanical capabilities of flapping robots [151]. However, applications remain limited because it has proven difficult for tailless flapping robots to control themselves via an onboard camera, given their limited payload capacity and aggressive vibration [136]. One work introduced a mechanical camera stabilization system, however, this introduces complexity and weight, and so

a software solution is preferred if possible [133, 152]. Recent work has shown that event cameras are the ideal sensor for these platforms, however the weight of currently available systems, and their need for a powerful onboard computer has made implementation challenging [131, 132]. Vision based control of a tailed flapping robot has been achieved with wireless monocular [127] and a specialized miniature stereo camera system [153]. However, rolling shutter effects remain a persistent challenge [133]. Recently, video stabilization has been shown to be beneficial for both frame and event based perception with a focus on flapping robots. However, the method relies on access to ground truth orientation [154]. Hence, the shaking of vision sensors remains a challenging problem for autonomous flapping wing robots.

Artificial Microsaccade Compensation produces the first stable, smooth videos from a rolling shutter camera mounted on a tailless ornithopter. The algorithm runs in real time and relies on a carefully selected combination of video stabilization techniques that are suited for the platform.

5.3 Methods

The algorithm has four parts. First, we describe the direct method for aligning frames, which is an inverse Lucas-Kanade method that assumes a rotational warp function and known image intrinsics. Next, we describe the computation of stable view parameters via filtering on the group of rotations $SO(3)$. Next, we describe the stabilization procedure, which uses a buffer of previous frames and the computed orientations to suppress high-frequency rotations in the image. Finally, a variant based on purposeful saccades is described which results in an efficient recursive update. A block diagram is given in Figure 5.1 while Figure 5.2 provides more detail.

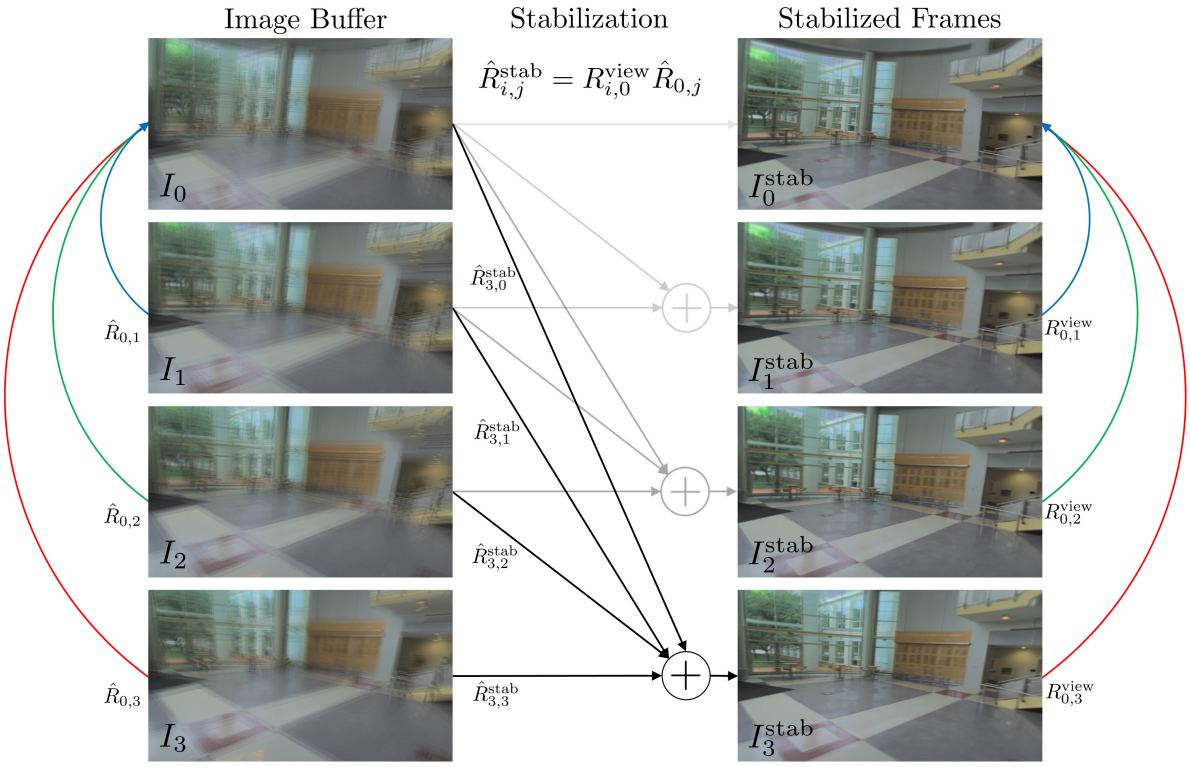

Figure 5.2: A detailed view of the "Stabilization" block in Figure 5.1. The left column is the average of 6 consecutive frames. While the individual frames are sharp, this averaging results in blur, which illustrates that the camera is shaking aggressively. The right column consists of the same six frames after stabilization and then averaging. In this paper, multiple frames are warped to the current stabilized view and averaged to reduce the effects of rolling shutter and transmission errors of a wireless camera onboard the Flapper. All displayed frames were computed using the camera's full resolution. Each timestep is associated with estimated orientations, $\hat{R}$ computed via a Lucas-Kanade tracker on the group of rotation matrices, $SO(3)$. Simultaneously, a list of stable orientations is computed R^{view}. The stabilized output frame at the current time is computed by warping current and previous frames to the current stabilized viewpoint using stabilizing rotations $\hat{R}^{\text{stab}}_{i,j} = R^{\text{view}}_{i,0}\hat{R}_{0,j}$.

5.3.1 Rotation Estimation

The Flapper has a mean flapping frequency of about 12 Hz, which causes the body and camera to oscillate with both translation and rotation at a similar frequency. However, the translational motion's contribution to optical flow is small in many applications. Thus, we propose ignoring the translational component of optical flow and only estimating the rotation under the assumption that the two frames are closely spaced in time. Formally, we assume the projection equations

$$p_c = \frac{1}{X_c^z} \begin{bmatrix} K & 0 \end{bmatrix} T_{c,w} X_w, \tag{5.1}$$

where p_c is the pixel corresponding to the projection of point X_w in the world frame into the camera frame, X_c^z is the depth at pixel p_c, $T_{c,w}$ is the $SE(3)$ transform from world to camera frame, and K is an invertible camera intrinsics matrix. In practice, the intrinsics of the fisheye lens used on the Flapper can be estimated beforehand. Thus, for the rest of the analysis, it is sufficient to assume $K = I$.

Then (5.1) can be expanded to consider the rotational and translation components of $T_{c,w}$ as $R_{c,w}$ and $t_{c,w}$ respectively,

$$p_c = \frac{1}{X_c^z} \left(R_{cw} X_w + t_{c,w} \right). \tag{5.2}$$

If it is assumed that $t_{c,w}$ can be considered zero, then the projection can be manipulated to reveal a relationship that does not depend on depth. Let $r_{c,w}^3$ be the third row of $R_{c,w}$ so then $X_c^z = r_{c,w}^3 X_w$. Also, define $p_w = X_w / X_w^z$, that is, the projection of world point X_w into the

image when the camera is at and aligned with the origin. Then

$$p_c = \frac{R_{c,w} X_w}{r^3_{c,w} X_w} = \frac{R_{cw} X_w / X^z_w}{r^3_{c,w} X_w / X^z_w} = \frac{R_{c,w} p_w}{r^3_{cw} p_w}. \tag{5.3}$$

Here we see that a world point's projection into the image after a rotation depends only on its starting pixel projection and that rotation. Further, this relationship is easily inverted. Then, if we define two camera frames c_k and c_j we find

$$\begin{aligned} p_{c_k} &= \frac{R_{c_k,w} p_w}{r^3_{c_k,w} p_w} \\ p_{c_j} &= \frac{R_{c_j,w} p_w}{r^3_{c_j,w} p_w} \\ \implies p_{c_k} &= \frac{R_{c_k,w} \left(r^3_{c_j,w} p_w \right) R_{w,c_j} p_{c_j}}{r^3_{c_k,w} \left(r^3_{c_j w} p_w \right) R_{w,c_j} p_{c_j}} \\ &= \frac{R_{c_k,c_j} p_{c_j}}{r^3_{c_k,c_j} p_{c_j}}. \end{aligned} \tag{5.4}$$

Thus, the correspondences between the pixel coordinates in frame c_k and c_j are obtained using just the rotation R_{c_k,c_j}. Notably, this works only because translation was neglected. Otherwise, an estimate or model of depth would be required to compute p_k from p_j. It is worth noting that homographies, also known as perspective warps, are commonly used to stabilize video feeds and generate panoramic images. However, homographies correspond to a planar depth assumption, which is well known to result in visible distortions when the scene is not planar. Thus, rotation is the only stabilizing effect that can be applied to an image without introducing distor-

tion via erroneous assumption or estimation of depth.

In what follows, it can be assumed that all considered coordinate frames are a camera frame at some time. Thus, for brevity, the notation $R_{k,j}$ will be used to indicate the rotation from frame c_j to c_k. Additionally, p_k and p_j will represent corresponding pixels in frames k and j taken from coordinate frames c_k and c_j. Further, all pixel coordinates are considered homogenous so that $(R_{k,j}p_j)/(r_{k,j}p_j)$ can be written as $R_{k,j}p_j$. That is, the division by the third element of $R_{k,j}p_j$ is assumed.

Then, we can set up an optimization problem to estimate $R_{k,j}$ from I_k and I_j by warping I_j so that the resulting pixel intensities are equal to those in I_k. The resulting problem is

$$\hat{R}_{k,j} = \underset{R_{k,j} \in SO(3)}{\operatorname{argmin}} \sum_{p_k \in I_k} \Big[I_k\big(p_k\big) - I_j\big(R_{j,k}p_k\big) \Big]^2. \tag{5.5}$$

We solve this problem using an inverse compositional Lucas-Kanade method [42] specialized for a rotational warp given known camera intrinsics K. The inverse compositional formulation iterates over the following two steps

$$\begin{aligned} \omega_{k,k-1} &= \underset{\omega \in so(3)}{\operatorname{argGN}} \sum_{p_k \in I_k} \Big[I_k\big(\exp(\omega)\, p_k\big) - I_j\big(R_{j,k-1}p_k\big) \Big]^2 \\ R_{j,k} &= R_{j,k-1} \exp(\omega_{k-1,k}) \end{aligned} \tag{5.6}$$

where argGN returns the result of a single step of a Gauss-Newton minimization and $R_{j,k}$ is used as $R_{j,k-1}$ in the next iteration. $\exp$ denotes the matrix exponential, which maps elements of $so(3)$ to $SO(3)$. The inverse compositional formulation is essential because each iteration can reuse precomputed gradients of I_k and the derivative of the rotational warp parameterized

by the $so(3)$ Lie Algebra which determine the approximate Hessian used by the Gauss-Newton minimization. These computations would otherwise be the most expensive computation of each iteration. To encourage stable tracking, we also incorporate a line search after each iteration. This guarantees that the loss decreases with each iteration, and prevents the estimate from diverging.

While the Lucas-Kanade method is well known, implementations of this variation are not widely available. Instead, most implementations consider the limited case of purely translational warps with coarse-to-fine refinement, as needed for tracking points. Additionally, some student implementations based on affine homographies are available, presumably because they are described in detail in [42]. However, as previously noted, affine homographies and full homographies correspond to an incorrect assumption on depth. Further, they require estimating more parameters (six or eight, respectively), while our method requires only three. We include our implementation, which is based on Jax, with this paper's open-source code.

It remains to track rotational changes across multiple frames. To do so, we propose the procedure described in Algorithm 1. In brief, the procedure estimates rotational deltas, $\hat{R}_{k,j}$, using the current frame I_j and a fixed template frame I_k using the previously described Lucas-Kanade method. Each optimization is warmstarted with the previous rotation estimate $\hat{R}_{k,j-1}$. The first template frame is the first frame I_0. After N_{track} frames, it is assumed that I_k should no longer be compared directly with the current frame I_j because of the compounding effects of the unmodeled translation. At such times, the template frame I_k is updated to be the current frame. Additionally, an estimate of the total rotation from time 0 to the new time k, called $\hat{R}_{0,k}$, is updated using the value of $\hat{R}_{k,j}$ at the time of the template reset. The process repeats indefinitely.

Reusing the template frame I_k for several iterations has several advantages. It limits drift in the cumulative orientation estimate because several new frames are compared to the same refer-

ence frame. Additionally, the expensive computation of the problem's gradients and approximate Hessian can be further postponed. While many criteria can be proposed to determine when to reset the template, such as the value of the residuals or the magnitude of the total rotation, we take the pragmatic approach of resetting every 5 frames.

5.3.2 Computation of a Stable View

Next, we consider the estimation of a stable viewpoint. That is, a coordinate frame that follows the estimated orientation but does not shake. In unconstrained problems, a standard approach would be to use a low pass filter of the estimated parameters. However, in this case, the estimated parameters are on $SO(3)$ and so a low-pass filter on $SO(3)$ is needed. For completeness, we describe our methodology, but note that a similar approach appeared in [147].

Let $R_{0,j}^{\text{view}}$ be the chosen stable viewpoint and recall that $\hat{R}_{0,j}$ is the estimate of the current cumulative orientation. Then the rotation from the current orientation to the stable viewpoint is

$$R_{j,j}^{\text{stab}} = R_{j,0}^{\text{view}} \hat{R}_{0,j} \tag{5.7}$$

because, by definition,

$$R_{0,j}^{\text{view}} R_{j,j}^{\text{stab}} = \hat{R}_{0,j}. \tag{5.8}$$

If $R_{0,j}^{\text{view}}$ follows $\hat{R}_{0,j}$, then $R_{j,j}^{\text{stab}}$ will be close to I. Thus, it is safe to assume $w_{j,j}^{\text{stab}} = \log(R_{j,j}^{\text{stab}}) \in so(3)$ exists and represents the geodesic direction and magnitude from $R_{0,j}^{\text{view}}$ to $\hat{R}_{0,j}$. This geodesic can then be used to interpolate between the two rotations according to

Algorithm 1 Artificial Microsaccade Compensation

Require: $N_{\text{track}}, N_{\text{avg}} \geq 0$

$I_k \leftarrow$ get_frame()

$\hat{R}_{0,k}, \hat{R}_{k,j}, R^{\text{view}}_{0,j} \leftarrow I$

$B_I, B_{\hat{R}}, B_{R^{\text{view}}} \leftarrow [\quad]$

$j \leftarrow 0$

while True **do**

/* Update Orientation Estimate */

$I_j \leftarrow$ get_frame()

$\hat{R}_{k,j} \leftarrow$ LK_inv_comp(I_k, I_j, $\hat{R}_{k,j}$)

$\hat{R}_{0,j} \leftarrow \hat{R}_{0,k}\hat{R}_{k,j}$

/* Update Stabilized View */

$\omega \leftarrow \log\left(R^{\text{view}}_{j,0}\hat{R}_{0,j}\right)$

$R^{\text{view}}_{0,j} \leftarrow R^{\text{view}}_{0,j} \exp\left(\alpha\left(\|\omega\|\right)\omega\right)$

/* Append to Buffers */

$B_I \leftarrow \text{append}(B_I, I_j)$

$B_{\hat{R}} \leftarrow \text{append}(B_{\hat{R}}, \hat{R})$

$B_{R^{\text{view}}} \leftarrow \text{append}(B_{R^{\text{view}}}, R^{\text{view}})$

/* Periodically Reset Template Image */

if $j \mod N_{\text{track}} = N_{\text{track}} - 1$ **then**

$I_k \leftarrow I_j$

$\hat{R}_{0,k} \leftarrow \hat{R}_{0,j}$

$\hat{R}_{k,j} \leftarrow I$

end if

/* Compute Stabilized Frame */

$B_{I^{\text{stab}}} \leftarrow [\quad]$

for $i \in \{j - N_{\text{avg}}, \ldots, j\}$ **do**

$\hat{R}^{\text{stab}}_{j,i} \leftarrow B_{R^{\text{view}}}[j]^T B_{\hat{R}}[i]$

$I^{\text{stab}}_{j,i} \leftarrow I^{\text{stab}}_j + \text{warp}(B_I[i], \hat{R}^{\text{stab}}_{j,i})$

$B_{I^{\text{stab}}} \leftarrow \text{append}(B_{I^{\text{stab}}}, I^{\text{stab}}_{j,i})$

end for

$I^{\text{stab}}_j \leftarrow \text{mean}\left(B_{I^{\text{stab}}}\right)$

$j \leftarrow j + 1$

end while

$$R^{\text{interp}}(\beta) = R_{0,j}^{\text{view}} \exp(\beta w_{j,j}^{\text{stab}}), \quad \beta \in [0,1]. \tag{5.9}$$

By definition, $R^{\text{interp}}(0) = R_{0,j}^{\text{view}}$ and $R^{\text{interp}}(1) = \hat{R}_{0,j}$. Suppose, $\beta \in (0,1)$, then a low pass filter on $SO(3)$ can be realized via the recursion

$$R_{0,j+1}^{\text{view}} = R_{0,j}^{\text{view}} \exp(\beta w_{j,j}^{\text{stab}}). \tag{5.10}$$

This filter is an analogue of a linear exponential filter. Such a filter may not converge fast enough to keep the stabilized view close to the true field of view, especially when the system is rotating quickly. We can improve the convergence at the expense of increasing the rotational noise by considering the magnitude of the rotation delta between the view and orientation estimate. Consider a monotonically increasing function $\alpha : \mathbb{R} \to \mathbb{R}$. Then the formulation can be generalized as

$$R_{0,j+1}^{\text{view}} = R_{0,j}^{\text{view}} \exp\left(\alpha\left(\|\omega_{j,j}^{\text{stab}}\|\right)\omega_{j,j}^{\text{stab}}\right) \tag{5.11}$$

In practice, we find the $\alpha(\|\omega\|) = a + b\|\omega\|$ with $a = 2\Delta t$, $b = 40\Delta t$, where Δt is the interframe period provides reasonable results.

5.3.3 Stabilization

At this point, the orientation of each frame has been estimated, $\hat{R}_{0,j}$ and a method for computing a stable viewpoint at each time, $R_{0,j}^{\text{view}}$, has been proposed. It remains to estimate a stabilized video. Consider the rotation delta going from a frame's estimated orientation at time i

to any stable viewpoint at time j, we call this $R_{j,i}^{\text{stab}}$, and it can be computed via

$$R_{j,i}^{\text{stab}} = R_{j,0}^{\text{view}} \hat{R}_{0,i}. \tag{5.12}$$

With this rotation estimate, we can warp any previous frame so that it appears to have been taken from the orientation of any stabilized viewpoint. We refer to such a rotation as a stabilizing rotation. The stabilized frames can be computed via

$$I_{j,i}^{\text{stab}}(p_j) = I_i(R_{i,j}^{\text{stab}} p_j). \tag{5.13}$$

If several such frames are computed for a particular time j, they can be combined, depending on the application, to produce a higher-quality stabilized frame. Specifically, in this paper, the stabilized frames are produced by averaging the last $N_{avg} = 6$ frames after aligning them to the current stabilized view. This reduces rolling shutter effects and transmission artifacts introduced by the wireless camera mounted on the flapping robot. That is, the final stabilized image is given by

$$I_j^{\text{stab}} = \frac{1}{N_{avg}} \sum_{i=j-N_{avg}}^{j} I_{j,i}^{\text{stab}} \tag{5.14}$$

5.3.4 Specialization to Saccades

The previously described stabilization method is useful for human viewing because the stabilized frames appear as if they were captured by a camera rotating smoothly. However, it is also possible to remove all rotation from an image for short periods of time by forcing $R_{0,j}^{\text{stab}}$ to

be constant until it needs to be changed to maintain overlap between the stabilized and true field of views. This results in the least possible apparent image motion because the stabilized frames become as identical as possible, until the static orientation is updated. Further, the stabilized frame can be computed via a recursion which saves significant computation. Consider, if $R^{\text{stab}}_{0,j} = R^{\text{stab}}_{0,j+1}$ then we can drop the subscripts from these terms, because they are identical, resulting in the recursion

$$\begin{aligned} I^{\text{stab}}_{j+1}(p_j) &= \frac{1}{N_{avg}} \sum_{i=j-N_{avg}+1}^{j+1} I_i(\hat{R}_{i,0} R^{\text{view}} p_j) \\ &= \frac{1}{N_{avg}} \Big[I^{\text{stab}}_j(p_j) + I_{j+1}(\hat{R}_{i,0} R^{\text{view}} p_j) \\ &\qquad - I_{j-N_{avg}}(\hat{R}_{i,0} R^{\text{view}} p_j) \Big] \end{aligned} \tag{5.15}$$

This expression adds and subtracts a frame from the previous stabilized frame to the get the next stabilized frame. In comparison, Equation 5.14 requires summing N_avg frames to produce a stabilized frame, which results in substantially more computation if $N_{avg} > 2$.

This recursion will produce suitable results until the true orientation of the camera rotates such that the true camera's field of view has little overlap with the stabilized image's field of view. At such times, the stabilized view must be shifted to align better with $\hat{R}_{0,j}$, similar to a saccade in human vision. For the purpose of study, when running in this recursive "saccade" mode, we reset R^{stab} to $\hat{R}_{0,j}$ when less than 90% of I^{stab}'s pixels are filled by each of the averaged frames.

5.4 Results and Discussion

The algorithm was tested on a Flapper Nimble+ manufactured by Flapper Drones. This tailless ornithopter beats its wings between approximately 12 and 20 Hz. The lack of a tail makes the platform agile, but also contributes to aggressive shaking. Our Flapper is equipped with a wireless HDZero Nano V3 FPV rolling shutter camera running at 1280×720 resolution and 60 fps. This camera is popular among racing drone pilots.

Videos showing stabilization results are available at `https://prg.cs.umd.edu/AMC`. These include comparison with Adobe Premier Pro's warp stabilizer, which is widely regarded as the best video stabilization software available. Adobe's documentation indicates that their method relies on feature matching and subspace warping [155]. In most sequences the video stabilized by Adobe Premier Pro is still vigorously shaking, and frequent distortions are introduced which make nearby objects look like they are stretching. In some sequences Adobe Premier Pro fails to produce a suitable result, with most of the frame remaining empty. The warp stabilizing plugin was configured with the maximum smoothness setting (100%) and with defaults for all other settings. On some sequences the software automatically advised disabling of cropping, which we did. It must be noted that the commercial software uses several additional gigabytes of RAM during stabilization, which is conducted in an offline, batch processing mode. Our method is capable of running online, achieves real-time performance, and uses several hundred megabytes total, most of which can be attributed to its research quality code which is written in Python and uses the high level JIT compilation and autodifferentiation framework Jax [156]. Specifically, autodifferentiation is used to precompute the static Jacobians used for every Gauss Newton step.

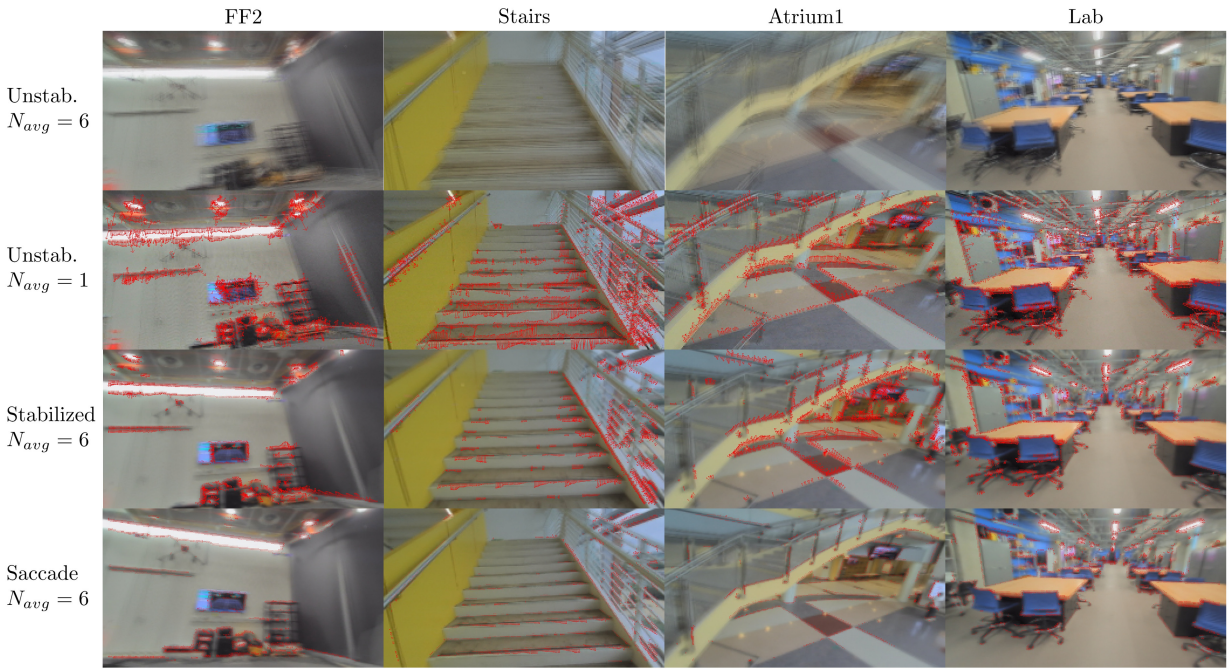

Figure 5.3: Qualitative comparison of stabilized and unstabilized frames. Top row, 6 averaged consecutive frames from a sequence illustrating the extreme shaking of a camera onboard the Flapper Nimble+ tailless ornithopter. Second row, a single unstabilized frame with normal flow (the projection of optical-flow onto gradients) illustrated with red arrows. Third row, 6 stabilized and averaged frames, which were warped to the stabilized viewpoint. Bottom row, 6 stabilized and averaged frames which were warped to a stationary viewpoint that occasionally sacccades. The flow magnitude can be seen to reduce with each row, with the saccading variation of the algorithm achieving a drastic reduction in flow magnitude. Videos showing stabilization results are available at `https://prg.cs.umd.edu/AMC`.

Figure 5.3 shows several qualitative results, showing a dramatic improvement of the image stability. In particular, the motion vectors from normal flow estimation, shown as red arrows, show that the apparent image motion is reduced drastically. Figure 5.4 shows per-frame metrics for sequence FF2. The metrics are described below. Equivalent figures for all collected sequences are available in the supplemental material. Tables 5.1 and 5.2 detail quantitative results averaged over each sequence.

For all results, the images were downsampled by 4x to 320×180 resolution. 12.5% margins were applied to each side of the output frames to limit the appearance of border effects due to

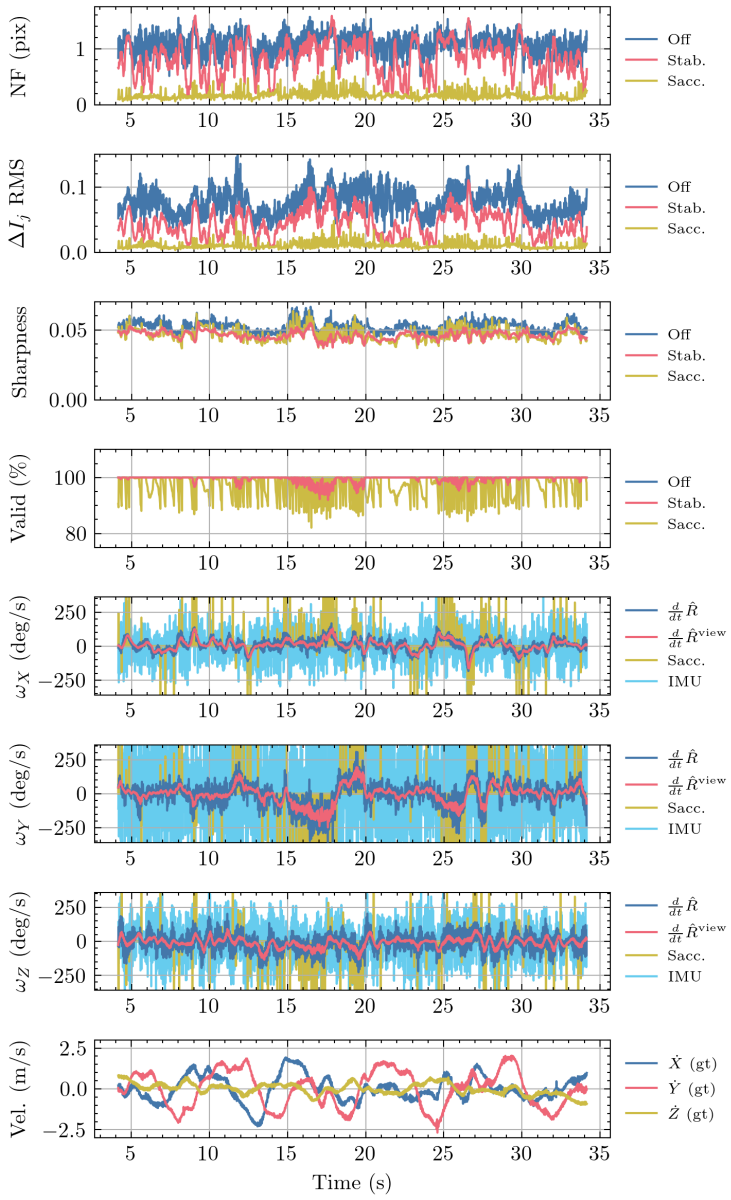

Figure 5.4: Per frame metrics computed on sequence FF2. Normal flow and the RMS change in image brightness are reduced by the stabilization and significantly reduced by saccading instead of continuously rotating the stable view. The proposed frame averaging, which reduces the effects of rolling shutter and image artifacts, slightly reduces the stabilized image sharpness. The stabilization algorithm maintains a high quantity of valid given 12.5% image margins. The saccading variant results in slightly fewer valid pixels. The angular velocities from the Flapper's onboard IMU do not align with the angular velocities of the image based orientation estimate. See Section 5.4.3 for a detailed discussion. The stabilized orientation does not contain the high-frequency oscillations of the estimated orientation. The saccade style stabilization maintains a viewpoint angular velocity of zero, except when it saccades, which results in an angular velocity spike.

Metric	Method	Sequence									
		UD	LR	FB	Yaw	Circle1	Circle2	FF1	FF2	FF3	Hand
NF RMS (pix.)	None	1.047	1.106	1.077	1.168	1.082	1.085	1.120	1.102	1.155	1.025
	Stab.	0.598	0.739	0.801	0.959	0.855	0.810	0.874	0.876	0.973	0.912
	Sacc.	0.129	0.155	0.157	0.207	0.169	0.175	0.182	0.174	0.197	0.103
ΔI_j RMS	None	0.075	0.080	0.070	0.089	0.073	0.076	0.083	0.079	0.080	0.078
	Stab.	0.032	0.041	0.038	0.062	0.049	0.040	0.049	0.048	0.053	0.062
	Sacc.	0.009	0.010	0.010	0.014	0.011	0.011	0.012	0.011	0.012	0.007
Sharpness	None	0.056	0.055	0.050	0.054	0.052	0.052	0.053	0.053	0.051	0.056
	Stab.	0.050	0.049	0.045	0.046	0.046	0.046	0.047	0.046	0.045	0.051
	Sacc.	0.050	0.048	0.045	0.048	0.047	0.045	0.047	0.047	0.046	0.051
Valid Pix. (%)	None	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	Stab	100.0	99.8	99.7	97.1	99.2	99.9	99.4	99.4	99.1	99.2
	Sacc.	97.8	97.4	97.6	96.8	97.2	97.4	97.2	97.1	97.2	96.7
ω RMS (deg/s)	MoCap	238	229	259	294	282	258	260	249	310	143
	IMU	301	289	268	368	347	276	308	311	325	90
	Image	85	97	88	157	128	93	107	105	124	92
ω^{view} RMS (deg/s)	Stab.	32	50	43	130	99	48	71	70	87	82
	Sacc.	21	44	39	129	97	42	66	65	89	75
V RMS (m/s)	MoCap	0.908	1.157	1.294	0.815	1.410	1.576	1.338	1.313	1.658	0.591

Table 5.1: Image quality, stabilization quality, angular velocity, and translational velocity metrics for ten sequences collected in a motion capture lab, each featuring different types of motion.

Metric	Method	Sequence				
		Hall	Stairs	Atrium1	Atrium2	Lab
NF RMS (pix.)	None	1.026	1.104	1.053	1.119	1.052
	Stab.	0.554	0.694	0.734	0.853	0.574
	Sacc.	0.229	0.231	0.170	0.192	0.203
ΔI_j RMS	None	0.048	0.073	0.078	0.079	0.078
	Stab.	0.018	0.029	0.043	0.048	0.033
	Sacc.	0.009	0.011	0.010	0.010	0.012
Sharpness	None	0.047	0.053	0.054	0.053	0.060
	Stab.	0.042	0.045	0.047	0.046	0.053
	Sacc.	0.041	0.045	0.047	0.046	0.052
Valid Pix. (%)	None	100.0	100.0	100.0	100.0	100.0
	Stab.	100.0	100.0	99.7	99.5	100.0
	Sacc.	97.3	98.0	97.7	97.4	97.6
ω RMS (deg/s)	IMU	250	273	273	276	234
	Image	80	99	101	110	80
ω^{view} RMS (deg/s)	Stab.	24	39	57	63	32
	Sacc.	17	32	49	58	22

Table 5.2: Image quality, stabilization quality, and angular velocity metrics for five sequences collected indoors.

the stabilized view's field of view being outside of the real camera's field of view. The final output resolution was 240×152. At this resolution, the algorithm achieved real-time performance as needed for future downstream applications. Specifically, on sequence Atrium2, it achieved 67 and 77 fps in the stabilized and saccade modes, respectively. Note that our code is implemented in Python, is almost entirely single threaded, loads images from disk in an uncompressed format, and does not use hardware acceleration for image interpolation which would speedup undistortion, the Lucas-Kanade tracker, and computation of I^{stab}. Figure 5.2 contains examples of high-resolution stabilized frames without downsampling.

5.4.1 Sequences

Ten sequences, each approximately 20 seconds in length, were collected in a motion capture room. The Flapper was piloted manually to achieve the following trajectories. Up-down (UD), left-right (LR), front-back (FB), yaw in place, fly in a circle (Circle1, Circle2), fly freely according to the operators discretion (FF1, FF2, FF3). The tenth sequence was collected by holding the flapper in hand, and quickly tilting it back and forth along each primary axis, followed by a smooth figure 8 motion. This in-hand trajectory is meant to demonstrate the best possible image quality when the camera's motion is smooth.

An additional five sequences, each approximately 20 seconds in length, were collected in a variety of indoor settings. Due to airspace restrictions, we were unable to collect data outside at the time of writing. The sequences include a hallway, stairwell, a large atrium (2x), and a robotics laboratory. These sequences test the algorithm with a variety of visual features and scene depths.

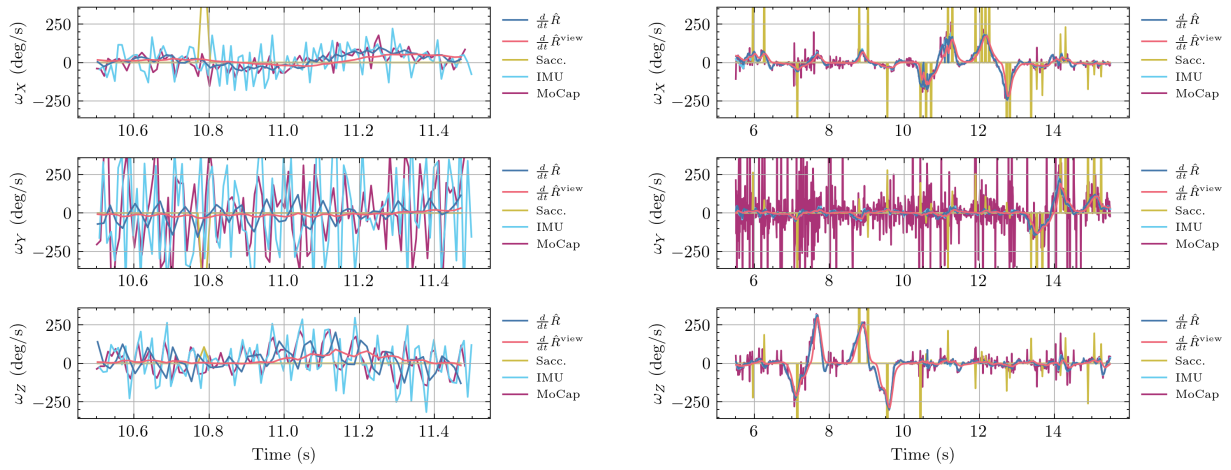

Figure 5.5: Zoomed in view of angular velocity estimates. **Left**: Angular velocity estimates from an onboard IMU, motion capture, the image orientation tracker, stabilized view, and saccade times during flight. The Flapper's onboard IMU fails to capture accurate enough angular velocity estimates for image stabilization during flight. **Right:** When moved smoothly by hand, the Flapper's onboard IMU's angular velocity estimates almost perfectly align with the image based angular velocity estimates. The motion capture system's angular velocity estimates also align, except for estimates along the Y axis, which are poor in quality because of occluded markers.

5.4.2 Metrics

We compute a variety of metrics to demonstrate the image stabilization quality. The first is the RMS difference between consecutive output images. As can be expected, the stabilized frames feature a noticeable reduction in RMS difference between subsequent frames, often achieving an approximately 50% reduction. Further, the saccade variant, which results in stabilized frames with a fixed orientation, achieves an approximately 7x reduction. These results can be expected because the Lucas-Kanade tracker minimizes the MSE between frames, though not necessarily pairs adjacent in time.

Next, we estimate image motion in the unstabilized, stabilized, and saccade stabilized frames using RMS normal flow magnitude. Normal flow has units of pixel displacement and has the magnitude of the optical flow onto the image gradients and can be estimated without the

assumptions required for feature matching. The normal flow at a pixel is given by

$$n(p) = -\nabla_t I(p) \frac{\nabla_p I(p)}{\|\nabla_p I(p)\|^2}. \tag{5.16}$$

The gradients are estimated using a Sobel filter with a kernel size of 3. To avoid erroneous estimation from gradients of small magnitude, pixels with gradients less than $15/255$ are discarded from the RMS normal flow magnitude calculation. The unstabilized frames experience flow magnitudes of approximately 1.1 pixels per frame. Stabilization reduces the normal flow magnitude to 0.5 to 0.9 pixels, however, it must be remembered that stabilization does not eliminate the measured rotational motion. Instead, it limits the high-frequency components of that motion. Finally, saccade-style stabilization reduces the normal flow magnitude to approximately 0.1 to 0.23 pixels.

Averaging frames results in a slight reduction in image sharpness. We measure image sharpness as the RMS magnitude of the image gradients estimated across all three color channels. While RMS contrast (or image variance) is a more typical measure of sharpness, it failed to quantify the reduction in sharpness due to frame averaging in this application. Our chosen measure indicates an approximately 10% reduction in the sharpness measure when $N_{avg} = 6$.

The average number of valid pixels, that is, pixels in I^{stab} which were filled by each of the averaged frames, is reported. The stabilized frames typically contain $> 99\%$ valid pixels when $N_{avg} = 6$. Saccade stabilization achieves a lower score, as expected, typically above $> 97\%$.

We report the average angular velocity estimated by motion capture, the Flapper's onboard IMU, image based orientation estimate, and the stabilized view. As will be discussed later, the IMU and motion capture angular velocity estimates appear to be relatively inaccurate during

flight, but accurate when the flapper is moved in-hand. We include them for completeness. The average angular velocity measured by the IMU is typically more than 250 deg/s and can be as high as 370 deg/s. Motion capture reports lower angular velocities ranging from 230 to about 300 degrees per second. The image-based orientation estimator reports much lower angular velocities between 80 and 160 deg/s. Notably, in the in-hand sequence, where the flapper was not shaking, the IMU and image based angular velocity estimates are almost identical. The stabilized view's average angular velocity is between 30 and 130 deg/s in all sequences and typically below 90 deg/s. Note that the saccade stabilized view's angular velocity is zero at all times except when a saccade occurs.

For the sequences recorded in the motion capture room, we report the flapper's average translational velocity which ranged between 0.9 and 1.7 m/s.

5.4.3 IMU Angular Velocity Estimates

The angular velocity estimates from the Flapper's onboard IMU are not accurate enough during flight to achieve an appreciable stabilization effect. This can be seen clearly from the left side of Figure 5.5 where the IMU estimates are not aligned with image based velocity estimates computed from $\hat{R}$. In particular, the estimates from the IMU have a much larger magnitude than those measured from the image. However, when the Flapper is moved smoothly, as in the Hand sequence, the IMU's angular velocities almost perfectly match with our image-based estimate as shown on the right side of Figure 5.5. The motion capture system also has difficulty estimating the Flapper's angular velocity. However, this can be expected because the Flapper's body is semi-flexible, so the motion capture markers do not move rigidly with respect to each other.

Determining the exact reason for the inaccuracy of the Flapper's IMU is the subject of future work. However, there are several possible explanations. The IMU is not mounted directly to the camera, instead it is on a nearby circuit board. Thus, the vibrations and semi-flexible body of the Flapper may cause the camera and IMU to move differently. Additionally, the Flapper's IMU may not be configured appropriately at the hardware level for measuring the Flapper's vibration patterns. Finally, it may be necessary to record the IMU at a rate higher than the maximum allowed by the Flapper's firmware, which is 100 Hz. Future work should investigate fusing gyro sensors with the proposed algorithm. However, the current results are acceptable for use.

5.4.4 Future Work

There are many directions for future work. Alternative methods for fusing frames, without reducing sharpness are of particular interest. Additionally, incorporating automatic identification of camera intrinsics would eliminate the need for prior calibration. Finally, while the Flapper's IMU was not helpful, it is of interest to more carefully study the use of IMU's for rough compensation during aggressive shaking. Finally, applications to robot dogs and humanoids are of interest because they also shake when walking and running.

5.5 Conclusion

Inspired by microsaccadic movements of the eye, and the human's inability to consciously perceive them, a specialized algorithm for video stabilization has been proposed which results in a direct image matching problem parameterized by 3D rotation. We call it "Artificial Microsaccade Compensation". The resulting orientation estimates admit multiple methods of computing a

stable viewpoint to which unstabilized frames can be rendered. The result is a video that appears to have been taken from a stable viewpoint, despite the real system shaking aggressively. The algorithm runs in real time at a reduced resolution, and there are many opportunities for optimization. The method was demonstrated on the Flapper Nimble+, a commercially available tailless ornithopter that has so far escaped camera-based sensing solutions due to the intense vibrations induced by its wings.

Chapter 6: Conclusion and Future Work

Embodied Representation has been developed starting from historically significant frameworks for perception and action discussed in Chapter 1. Technical preliminaries for tight coupling of vision and action, through a bio-inspired perceptual visual representation called time-to-contact was established in Chapter 2. Subsequently, Chapter 3 developed a general mathematical formulation for Embodied Representation, resulting in two specific algorithms for fundamental problems that were unsolved in the robotics literature. That is, uncalibrated clearing and jumping. That is, a robot with no initial knowledge of its size, strength, strength of gravity, or the size of the world became able to determine if it fits through an opening before reaching it and jumping a gap on the first attempt. Subsequently, Chapter 4 provided an introduction to the use of Embodied Representation and internal feedback for tactile manipulation, resulting in a simple algorithm, with just a handful of parameters, that could solve a challenging key-in-lock insertion problem and outperform a sophisticated reinforcement learning baseline trained on the objects. Finally, Chapter 5 addressed an important sub-problem in the control bio-inspired robots, image stabilization under aggressive shaking, and achieves the first stable video from a tailless ornithopter by representing visual inputs in a different coordinate frame than the body.

There are many additional applications and specific theories that can be developed through Embodied Representation's mathematical framework. Further, because of its natural robustness

properties, the theory has important implications for practical application. Its development may result in a paradigm shift towards using embodied representations that realize robust, adaptive visuomotor behavior in broad classes of robots. In what follows we outline the most important high-level directions that will popularize the theory and encourage broad impact. At a high level, they are:

- Target interception with an uncalibrated drone
- Throwing with an uncalibrated robot arm
- Visuo-tactile manipulation of unknown objects
- Image-based and qualitative visuomotor representation

6.1 Future Work

6.1.1 Uncalibrated Target Interception

An extension of Embodied Representation's theory allows the pursuit of moving objects by adding unknown parameters of projectile motion to the robot's motion model. However, the development of a satisfactory controller for the problem is difficult. Time delays due to communication and image processing can approach 10-100 milliseconds in a typical drone. Dragonflies are thought to solve the challenge of an approximately 50-millisecond delay with predictive processing [55]. Since Embodied Representation admits similar predictive processing and development of the self-model required to fly an uncalibrated drone, it is of interest to attempt the challenge of autonomously flying a drone through a moving hoop using the onboard camera. If realized,

the solution will bear similarity to human pilots who can quickly learn to aggressively fly a drone despite imprecise controls and significant time delays.

6.1.2 Uncalibrated Throwing

Developing an Embodied Representation for a robotic arm is also of interest. Robot arms are typically manufactured precisely, which adds to their cost. However, many mammals make precise use of their arms, which change throughout their lifespan, without ever being externally calibrated. An adaptive approach built on Embodied Representation may admit a path toward using inexpensive and consequently imprecise robot arms in applications typically reserved for high-end variants. In particular, it is of interest to study the application of throwing, which is still a difficult task even for calibrated robot arms [157].

6.1.3 Uncalibrated Manipulation

Manipulation is a particularly difficult problem in robotics, where many limitations stem from reliance on precise modeling of contact. It is expected that Embodied Representation may help with such problems because it is designed to develop models online from visuomotor feedback. Further, it is particularly suited for use with tactile sensors like the GelSight Mini that produce an image. While Chapter 4 touches on this topic for the application of key insertion and achieves great success, there is much more to accomplish. Future work should extend this formulation to more carefully consider Embodied Representation and generalize it to vision guided insertion tasks such as routing a cable through a hole and inserting cups into a crowded cupboard.

6.1.4 Image-Based and Qualitative Embodied Representation

Finally, it is also of interest to further develop the basic mathematical theory of Embodied Representation. In particular, it is thought that humans rely on image-based and qualitative representations of position and distance when driving a car and catching high-flying objects; that is, they do not primarily rely on 3D position [13, 82]. While similar theories have been developed in robotics, they typically lack any stability guarantees and rely on control loops adapted for particular operating conditions [31]. If developed further, the sense of embodied distance afforded by Embodied Representation may alleviate these issues, thus making these image-based and qualitative approaches broadly applicable in robotics practice.

6.2 Conclusion

To conclude, suppose you are at your desk looking at some objects on it. You don't know the precise distance from your eye to any particular object in meters. However, you can immediately reach out and touch any of them. Instead of the meter, your knowledge of distance is encoded in unknown but embodied units of action. Realizing this general ability in robots is what Embodied Representation aims to accomplish. The result will have a broad impact on robotics theory and practice because it is a pre-requisite for uncalibrated, imprecisely made robots that adapt to new conditions and tasks, much like animals.

Appendix A: Permissions

A.1 IEEE Copyright Notice

In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of University of Maryland's products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to `http://www.ieee.org/publications_standards/publications/rights/rights_link.html` to learn how to obtain a License from RightsLink.

Specifically, the IEEE copyrighted material is Chapter 2 titled "TTCDist: Fast Distance Estimation From an Active Monocular Camera Using Time-to-Contact" and Figure 1.2.

A.2 Creative Commons License

Chapter 3 titled "Embodied Visuomotor Representation" reproduces [158] published by Springer Nature. It is reproduced under the Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format. No changes were made beyond inserting links in figure caption's to supplemental videos

that are available online. A copy of the license can be viewed at: http://creativecommons.org/licenses/by/4.0/.

Bibliography

[1] J von Uexküll. Theoretical biology. 1926.

[2] Joaquin M Fuster. The prefrontal cortex makes the brain a preadaptive system. *Proceedings of the IEEE*, 102(4):417–426, 2014.

[3] HLF von Helmholtz. Treatise on physiological optics, 3 vols. 1924.

[4] David Marr. *Vision: A computational investigation into the human representation and processing of visual information*. MIT press, 2010.

[5] John Aloimonos, Isaac Weiss, and Amit Bandyopadhyay. Active vision. *International Journal of Computer Vision*, 1(4):333–356, 1988.

[6] Ruzena Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, 1988.

[7] Dana H. Ballard. Animate vision. *Artificial Intelligence*, 48(1):57–86, 1991.

[8] Dana H. Ballard and Christopher M. Brown. Principles of animate vision. *CVGIP: Image Understanding*, 56(1):3–21, 1992.

[9] David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. Ieee, 1999.

[10] Hugh Durrant-Whyte, David Rye, and Eduardo Nebot. Localization of autonomous guided vehicles. In *Robotics Research: The Seventh International Symposium*, pages 613–625. Springer, 1996.

[11] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, et al. Stanley: The robot that won the darpa grand challenge. *Journal of field Robotics*, 23(9):661–692, 2006.

[12] Steven M LaValle and James J Kuffner Jr. Randomized kinodynamic planning. *The international journal of robotics research*, 20(5):378–400, 2001.

[13] David N. Lee. A theory of visual control of braking based on information about time-to-collision. *Perception*, 5(4):437–459, 1976.

[14] Sridhar Ravi, Tim Siesenop, Olivier Bertrand, Liang Li, Charlotte Doussot, William H Warren, Stacey A Combes, and Martin Egelhaaf. Bumblebees perceive the spatial layout of their environment in relation to their body size and form to minimize inflight collisions. *Proceedings of the National Academy of Sciences*, 117(49):31494–31499, 2020.

[15] Colin G Ellard, Melvyn A Goodale, and Brian Timney. Distance estimation in the mongolian gerbil: The role of dynamic depth cues. *Behavioural brain research*, 14(1):29–39, 1984.

[16] Cornelia Fermüller. Navigational preliminaries. In Yiannis Aloimonos, editor, *Active Perception*, chapter 3, pages 103–150. Lawrence Erlbaum Assoc., Hillsdale, NJ, 1993.

[17] Cornelia Fermüller and Yiannis Aloimonos. Vision and action. *Image and Vision Computing*, 13(10):725–744, 1995.

[18] Nitin Jagannatha Sanket. *Active Vision Based Embodied-AI Design for Nano-UAV Autonomy*. PhD thesis, University of Maryland, College Park, 2021.

[19] Ruzena Bajcsy, Yiannis Aloimonos, and John K. Tsotsos. Revisiting active perception. *Autonomous Robots*, 42:177–196, 2018.

[20] Cornelia Fermüller and Yiannis Aloimonos. Tracking facilitates 3-D motion estimation. *Biological Cybernetics*, 67(3):259–268, 1992.

[21] Ajay Mishra, Yiannis Aloimonos, and Cornelia Fermüller. Active segmentation for robotics. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3133–3139. IEEE, 2009.

[22] Vadim Indelman, Stephen Williams, Michael Kaess, and Frank Dellaert. Information fusion in navigation systems via factor graph based incremental smoothing. *Robotics and Autonomous Systems*, 61(8):721–738, 2013.

[23] Fendy Santoso, Matthew A. Garratt, and Sreenatha G. Anavatti. Visual–inertial navigation systems for aerial robotics: Sensor fusion and technology. *IEEE Transactions on Automation Science and Engineering*, 14(1):260–275, 2017.

[24] Tong Qin, Peiliang Li, and Shaojie Shen. VINS-Mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.

[25] Michael Bloesch, Sammy Omari, Marco Hutter, and Roland Siegwart. Robust visual inertial odometry using a direct EKF-based approach. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 298–304, 2015.

[26] Maximilian Krogius, Acshi Haggenmiller, and Edwin Olson. Flexible layouts for fiducial tags. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1898–1903, 2019.

[27] David N Lee, Reinoud J Bootsma, Mike Land, David Regan, and Rob Gray. Lee's 1976 paper. *Perception*, 38(6):837–858, 2009.

[28] Olaf Sikorski, Dario Izzo, and Gabriele Meoni. Event-based spacecraft landing using time-to-contact. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 1941–1950, 2021.

[29] Celyn Walters and Simon Hadfield. EVReflex: Dense time-to-impact prediction for event-based obstacle avoidance. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1304–1309, 2021.

[30] Hann Woei Ho, Guido CHE de Croon, and Qiping Chu. Distance and velocity estimation using optical flow from a monocular camera. *International Journal of Micro Air Vehicles*, 9(3):198–208, 2017.

[31] Guido C H E de Croon. Monocular distance estimation with optical flow maneuvers and efference copies: a stability-based strategy. *Bioinspiration & Biomimetics*, 11(1):016004, 2016.

[32] Abhishek Badki, Orazio Gallo, Jan Kautz, and Pradeep Sen. Binary TTC: A temporal geofence for autonomous navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12946–12955, 2021.

[33] Ziyun Wang, Fernando Cladera Ojeda, Anthony Bisulco, Daewon Lee, Camillo J. Taylor, Kostas Daniilidis, M. Ani Hsieh, Daniel D. Lee, and Volkan Isler. EV-Catcher: High-speed object catching using low-latency event-based neural networks. *IEEE Robotics and Automation Letters*, 7(4):8737–8744, 2022.

[34] Berthold K.P. Horn, Yajun Fang, and Ichiro Masaki. Time to contact relative to a planar surface. In *2007 IEEE Intelligent Vehicles Symposium*, pages 68–74, 2007.

[35] Anastasios I. Mourikis and Stergios I. Roumeliotis. A multi-state constraint Kalman filter for vision-aided inertial navigation. In *Proceedings of the 2007 IEEE International Conference on Robotics and Automation*, pages 3565–3572, 2007.

[36] Agostino Martinelli. Vision and IMU data fusion: Closed-form solutions for attitude, speed, absolute scale, and bias determination. *IEEE Transactions on Robotics*, 28(1):44–60, 2012.

[37] Agostino Martinelli. Closed-form solution of visual-inertial structure from motion. *International Journal of Computer Vision*, 106(2):138–152, 2014.

[38] Ronald Clark, Sen Wang, Hongkai Wen, Andrew Markham, and Niki Trigoni. VINet: Visual-inertial odometry as a sequence-to-sequence learning problem. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, volume 31, pages 3995–4001, 2017.

[39] Liming Han, Yimin Lin, Guoguang Du, and Shiguo Lian. DeepVIO: Self-supervised deep learning of monocular visual inertial odometry using 3D geometric constraints. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6906–6913. IEEE, 2019.

[40] Nitin J Sanket, Chahat Deep Singh, Cornelia Fermüller, and Yiannis Aloimonos. PRGFlow: Unified SWAP-aware deep global optical flow for aerial robot navigation. *Electronics Letters*, 57(16):614–617, 2021.

[41] João P. Hespanha. *Linear Systems Theory*. Princeton University Press, 2nd edition, 2018.

[42] Simon Baker and Iain Matthews. Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, 2004.

[43] Mikhail Sh, Birman, and Michael Solomyak. Lectures on double operator integrals, 2003.

[44] Leonid Keselman, John Iselin Woodfill, Anders Grunnet-Jepsen, and Achintya Bhowmik. Intel(R) RealSense(TM) stereoscopic depth cameras. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1267–1276, 2017.

[45] Siu Kwan Lam, Antoine Pitrou, and Stanley Seibert. Numba: A LLVM-based Python JIT compiler. In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, pages 7:1–7:6, 2015.

[46] Michael Bloesch, Michael Burri, Sammy Omari, Marco Hutter, and Roland Siegwart. Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback. *The International Journal of Robotics Research*, 36(10):1053–1072, 2017.

[47] Zichao Zhang and Davide Scaramuzza. A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7244–7251, 2018.

[48] Dana H. Ballard. Animate vision. *Artificial Intelligence*, 48(1):57–86, 1991.

[49] Arthur G. Stephenson, Lia S. LaPiana, Daniel R. Mulville, Peter J. Rutledge, Frank H. Bauer, David Folta, Greg A. Dukeman, Robert Sackheim, Peter Norvig, Edward J. Weiler, and Frederick D. Gregory. *Mars Climate Orbiter Mishap Investigation Board Phase I Report*. National Aeronautics and Space Administration, 1999.

[50] Jerome A Feldman. Four frames suffice: A provisional model of vision and space. *Behavioral and Brain Sciences*, 8(2):265–289, 1985.

[51] Jan J Koenderink, Andrea J Van Doorn, and Astrid ML Kappers. Surface perception in pictures. *Perception & Psychophysics*, 52:487–496, 1992.

[52] James T Todd. The visual perception of 3D shape. *Trends in cognitive sciences*, 8(3):115–121, 2004.

[53] LoongFah Cheong, Cornelia Fermüller, and Yiannis Aloimonos. Effects of errors in the viewing geometry on shape estimation. *Computer Vision and Image Understanding*, 71(3):356–372, 1998.

[54] Hui Ji and Cornelia Fermüller. Noise causes slant underestimation in stereo and motion. *Vision Research*, 46(19):3105–3120, 2006.

[55] Matteo Mischiati, Huai-Ti Lin, Paul Herold, Elliot Imler, Robert Olberg, and Anthony Leonardo. Internal models direct dragonfly interception steering. *Nature*, 517(7534):333–338, 2015.

[56] Franck Ruffier and Nicolas Franceschini. Optic flow regulation: the key to aircraft automatic guidance. *Robotics and Autonomous Systems*, 50(4):177–194, 2005.

[57] Francois Chaumette and Seth Hutchinson. Visual servo control. I. Basic approaches. *IEEE Robotics & Automation Magazine*, 13(4):82–90, 2006.

[58] Francois Chaumette and Seth Hutchinson. Visual servo control. II. Advanced approaches. *IEEE Robotics & Automation Magazine*, 14(1):109–118, 2007.

[59] Bruno Herissé, Tarek Hamel, Robert Mahony, and François-Xavier Russotto. Landing a VTOL unmanned aerial vehicle on a moving platform using optical flow. *IEEE Transactions on Robotics*, 28(1):77–89, 2012.

[60] Guido CHE De Croon, Julien JG Dupeyroux, Christophe De Wagter, Abhishek Chatterjee, Diana A Olejnik, and Franck Ruffier. Accommodating unobservability to control flight attitude with optic flow. *Nature*, 610(7932):485–490, 2022.

[61] E. Malis, F. Chaumette, and S. Boudet. 2 1/2 D visual servoing. *IEEE Transactions on Robotics and Automation*, 15(2):238–250, 1999.

[62] P.I. Corke and S.A. Hutchinson. A new partitioned approach to image-based visual servo control. *IEEE Transactions on Robotics and Automation*, 17(4):507–515, 2001.

[63] Arthur M. Glenberg. What memory is for. *Behavioral and Brain Sciences*, 20(1):1–19, 1997.

[64] Oliver Brock. Intelligence as computation. *IOP Conference Series: Materials Science and Engineering*, 1321(1):012001, dec 2024.

[65] Daniel McNamee and Daniel M. Wolpert. Internal models in biological control. *Annual Review of Control, Robotics, and Autonomous Systems*, 2(1):339–364, 2019.

[66] J. Huang, A. Isidori, L. Marconi, M. Mischiati, E. Sontag, and W. M. Wonham. Internal models in control, biology and neuroscience. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 5370–5390, 2018.

[67] K.J. Åström and B. Wittenmark. *Adaptive Control*. Dover Publications, 2008.

[68] Levi Burner, Nitin J. Sanket, Cornelia Fermüller, and Yiannis Aloimonos. TTCDist: Fast distance estimation from an active monocular camera using time-to-contact. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4909–4915, 2023. © 2023 IEEE. Reprinted, with permission, from Burner, Levi et al. "TTCDist: Fast Distance Estimation From an Active Monocular Camera Using Time-to-Contact", 2023 IEEE International Conference on Robotics and Automation (ICRA), May 2023.

[69] Daniel Raviv. *A quantitative approach to looming*. US Department of Commerce, National Institute of Standards and Technology, 1992.

[70] Dario Izzo and Guido Croon. Nonlinear model predictive control applied to vision-based spacecraft landing. In *Proceedings of the EuroGNC 2013, 2nd CEAS Specialist Conference on Guidance, Navigation & Control*, pages 91–107, 2013.

[71] Nicolas Franceschini, Franck Ruffier, and Julien Serres. A bio-inspired flying robot sheds light on insect piloting abilities. *Current Biology*, 17(4):329–335, 2007.

[72] Julien Serres, D Dray, Franck Ruffier, and N Franceschini. A vision-based autopilot for a miniature air vehicle: joint speed control and lateral obstacle avoidance. *Autonomous robots*, 25:103–122, 2008.

[73] Franck Ruffier and Nicolas Franceschini. Optic flow regulation in unsteady environments: A tethered mav achieves terrain following and targeted landing over a moving platform. *Journal of Intelligent & Robotic Systems*, 79:275–293, 2015.

[74] Jian Chen, D.M. Dawson, W.E. Dixon, and A. Behal. Adaptive homography-based visual servo tracking for a fixed camera configuration with a camera-in-hand extension. *IEEE Transactions on Control Systems Technology*, 13(5):814–825, 2005.

[75] Daniel M. Wolpert, Zoubin Ghahramani, and Michael I. Jordan. An internal model for sensorimotor integration. *Science*, 269(5232):1880–1882, 1995.

[76] Boyuan Chen, Robert Kwiatkowski, Carl Vondrick, and Hod Lipson. Fully body visual self-modeling of robot morphologies. *Science Robotics*, 7(68):eabn1944, 2022.

[77] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[78] Dhruv Shah, Benjamin Eysenbach, Gregory Kahn, Nicholas Rhinehart, and Sergey Levine. ViNG: Learning open-world navigation with visual goals. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13215–13222, 2021.

[79] Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. GNM: A general navigation model to drive any robot. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7226–7233, 2023.

[80] Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. ViNT: A foundation model for visual navigation. In *2023 Conference on Robot Learning (CoRL)*, 2023.

[81] Mauro Bisiacco and Gianluigi Pillonetto. On the mathematical foundations of stable RKHSs. *Automatica*, 118:109038, 2020.

[82] Seville Chapman. Catching a baseball. *American Journal of Physics*, 36(10):868–870, 10 1968.

[83] D. Regan. Visual factors in hitting and catching. *Journal of Sports Sciences*, 15(6):533–558, 1997.

[84] Mathieu Blondel, Quentin Berthet, Marco Cuturi, Roy Frostig, Stephan Hoyer, Felipe Llinares-López, Fabian Pedregosa, and Jean-Philippe Vert. Efficient and modular implicit differentiation. *Advances in neural information processing systems*, 35:5230–5242, 2022.

[85] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012.

[86] M. S. Andersen, J. Dahl, Z. Liu, and L. Vandenberghe. Interior-point methods for large-scale cone programming. In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*, pages 55–83. MIT Press, 2012.

[87] Sangwoon Kim and Alberto Rodriguez. Active extrinsic contact sensing: Application to general peg-in-hole insertion. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 10241–10247, 2022.

[88] Bingjie Tang, Iretiayo Akinola, Jie Xu, Bowen Wen, Ankur Handa, Karl Van Wyk, Dieter Fox, Gaurav S. Sukhatme, Fabio Ramos, and Yashraj Narang. AutoMate: Specialist and generalist assembly policies over diverse geometries. In *Proceedings of Robotics: Science and Systems*, Los Angeles, United States, June 2025.

[89] Tatsuya Kamijo, Ixchel G Ramirez-Alpizar, Enrique Coronado, and Gentiane Venture. Tactile-based active inference for force-controlled peg-in-hole insertions. *arXiv preprint arXiv:2309.15681*, 2023.

[90] Chaojie Yan, Jun Wu, and Qiuguo Zhu. Learning-based contact status recognition for peg-in-hole assembly. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6003–6009, 2021.

[91] Yansong Wu, Zongxie Chen, Fan Wu, Lingyun Chen, Liding Zhang, Zhenshan Bing, Abdalla Swikir, Sami Haddadin, and Alois Knoll. TacDiffusion: Force-domain diffusion policy for precise tactile manipulation. *arXiv preprint arXiv:2409.11047*, 2024.

[92] Wenzhen Yuan, Siyuan Dong, and Edward H. Adelson. GelSight: High-resolution robot tactile sensors for estimating geometry and force. *Sensors*, 17(12), 2017.

[93] Maurice Leblanc. Sur l'electrification des chemins de fer au moyen de courants alternatifs de frequence elevee. *Revue générale de l'électricité*, 12(8):275–277, 1922.

[94] Alexander Scheinker. 100 years of extremum seeking: A survey. *Automatica*, 161:111481, 2024.

[95] Anne Auger and Nikolaus Hansen. Tutorial CMA-ES: evolution strategies and covariance matrix adaptation. In *Proceedings of the 14th annual conference companion on Genetic and evolutionary computation*, pages 827–848, 2012.

[96] Hyeonjun Park, Jaeheung Park, Dong-Hyuk Lee, Jae-Han Park, Moon-Hong Baeg, and Ji-Hun Bae. Compliance-based robotic peg-in-hole assembly strategy without force feedback. *IEEE Transactions on Industrial Electronics*, 64(8):6299–6309, 2017.

[97] Hyeonjun Park, Jaeheung Park, Dong-Hyuk Lee, Jae-Han Park, and Ji-Hun Bae. Compliant peg-in-hole assembly using partial spiral force trajectory with tilted peg posture. *IEEE Robotics and Automation Letters*, 5(3):4447–4454, 2020.

[98] Oliver Gibbons, Alessandro Albini, and Perla Maiolino. A tactile feedback insertion strategy for peg-in-hole tasks. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10415–10421, 2023.

[99] Korbinian Nottensteiner, Freek Stulp, and Alin Albu-Schäffer. Robust, locally guided peg-in-hole using impedance-controlled robots. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5771–5777, 2020.

[100] Siyuan Dong and Alberto Rodriguez. Tactile-based insertion for dense box-packing. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7953–7960. IEEE, 2019.

[101] Rui Li, Robert Platt, Wenzhen Yuan, Andreas Ten Pas, Nathan Roscup, Mandayam A Srinivasan, and Edward Adelson. Localization and manipulation of small parts using gelsight tactile sensing. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3988–3993. IEEE, 2014.

[102] Yuni Fuchioka, Cristian C. Beltran-Hernandez, Hai Nguyen, and Masashi Hamaya. Robotic object insertion with a soft wrist through sim-to-real privileged training. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9159–9166, 2024.

[103] Yiting Chen, Kenneth Kimble, Howard H. Qian, Podshara Chanrungmaneekul, Robert Seney, and Kaiyu Hang. Robust peg-in-hole assembly under uncertainties via compliant and interactive contact-rich manipulation. In *Proceedings of Robotics: Science and Systems*, Los Angeles, United States, June 2025.

[104] Siyuan Dong, Devesh K. Jha, Diego Romeres, Sangwoon Kim, Daniel Nikovski, and Alberto Rodriguez. Tactile-RL for insertion: Generalization to objects of unknown geometry. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6437–6443, 2021.

[105] Sagar Gubbi, Shishir Kolathaya, and Bharadwaj Amrutur. Imitation learning for high precision peg-in-hole tasks. In *2020 6th International Conference on Control, Automation and Robotics (ICCAR)*, pages 368–372, 2020.

[106] Yansong Wu, Fan Wu, Lingyun Chen, Kejia Chen, Samuel Schneider, Lars Johannsmeier, Zhenshan Bing, Fares J. Abu-Dakka, Alois Knoll, and Sami Haddadin. 1 khz behavior tree for self-adaptable tactile insertion. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16002–16008, 2024.

[107] Yu She, Shaoxiong Wang, Siyuan Dong, Neha Sunil, Alberto Rodriguez, and Edward Adelson. Cable manipulation with a tactile-reactive gripper. *The International Journal of Robotics Research*, 40(12-14):1385–1401, 2021.

[108] R. J. Chang, C. Lin, and P. Lin. Visual-based automation of peg-in-hole microassembly process. *Journal of Manufacturing Science and Engineering*, 133:041015, 08 2011.

[109] Song Liu, De Xu, Fangfang Liu, Dapeng Zhang, and Zhengtao Zhang. Relative pose estimation for alignment of long cylindrical components based on microscopic vision. *IEEE/ASME Transactions on Mechatronics*, 21(3):1388–1398, 2016.

[110] Haeseong Lee, Suhan Park, Keunwoo Jang, Seungyeon Kim, and Jaeheung Park. Contact state estimation for peg-in-hole assembly using gaussian mixture model. *IEEE Robotics and Automation Letters*, 7(2):3349–3356, 2022.

[111] Kuangen Zhang, MinHui Shi, Jing Xu, Feng Liu, and Ken Chen. Force control for a rigid dual peg-in-hole assembly. *Assembly Automation*, 37:200–207, 04 2017.

[112] Haohong Lin, Radu Corcodel, and Ding Zhao. Generalize by touching: Tactile ensemble skill transfer for robotic furniture assembly. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9227–9233, 2024.

[113] Yijie Guo, Bingjie Tang, Iretiayo Akinola, Dieter Fox, Abhishek Gupta, and Yashraj Narang. SRSA: Skill retrieval and adaptation for robotic assembly tasks. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu, editors, *International Conference on Representation Learning*, volume 2025, pages 17082–17111, 2025.

[114] Isidoros Marougkas, Dhruv Metha Ramesh, Joe H Doerr, Edgar Granados, Aravind Sivaramakrishnan, Abdeslam Boularias, and Kostas E Bekris. Integrating model-based control and RL for sim2real transfer of tight insertion policies. *arXiv preprint arXiv:2505.11858*, 2025.

[115] Guanghe Li, Junming Zhao, Shengjie Wang, and Yang Gao. EasyInsert: A data-efficient and generalizable insertion policy. *arXiv preprint arXiv:2505.16187*, 2025.

[116] Naoki Fukaya, Koki Yamane, Shimpei Masuda, Avinash Ummadisingu, Shin-ichi Maeda, and Kuniyuki Takahashi. Four-axis adaptive fingers hand for object insertion: Faaf hand. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 13996–14003, 2024.

[117] Gaozhao Wang, Xing Liu, Zhengxiong Liu, Panfeng Huang, and Yang Yang. Visual-tactile perception based control strategy for complex robot peg-in-hole process via topological and geometric reasoning. *IEEE Robotics and Automation Letters*, 9(10):8410–8417, 2024.

[118] S.R. Chhatpar and M.S. Branicky. Localization for robotic assemblies using probing and particle filtering. In *Proceedings, 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics.*, pages 1379–1384, 2005.

[119] Zifan Zhao, Siddhant Haldar, Jinda Cui, Lerrel Pinto, and Raunaq Bhirangi. Touch begins where vision ends: Generalizable policies for contact-rich manipulation. *arXiv preprint arXiv:2506.13762*, 2025.

[120] Berk Calli, Wouter Caarls, Martijn Wisse, and Pieter P. Jonker. Active vision via extremum seeking for robots in unstructured environments: Applications in object recognition and manipulation. *IEEE Transactions on Automation Science and Engineering*, 15(4):1810–1822, 2018.

[121] Simon Baker, Ankur Datta, and Takeo Kanade. Parameterizing homographies. Technical Report CMU-RI-TR-06-11, Carnegie Mellon University, Pittsburgh, PA, March 2006.

[122] Jiawei Zhang, Chengchao Bai, and Jifeng Guo. Multiple peg-in-hole assembly of tightly coupled multi-manipulator using learning-based visual servo. *arXiv preprint arXiv:2407.10570*, 2024.

[123] Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. FoundationPose: Unified 6d pose estimation and tracking of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17868–17879, 2024.

[124] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. SAM 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024.

[125] Guido de Croon. Flapping wing drones show off their skills. *Science Robotics*, 5(44):eabd0233, 2020.

[126] Joon Hyuk Park and Kwang-Joon Yoon. Designing a biomimetic ornithopter capable of sustained and controlled flight. *J. Bionic Eng.*, 5(1):39–47, March 2008.

[127] GCHE De Croon, KME De Clercq, Remes Ruijsink, Bart Remes, and Christophe De Wagter. Design, aerodynamics, and vision-based control of the delfly. *International Journal of Micro Air Vehicles*, 1(2):71–97, 2009.

[128] Peng Nian, Bifeng Song, Jianlin Xuan, Wenhui Zhou, and Dong Xue. Study on flexible flapping wings with three dimensional asymmetric passive deformation in a flapping cycle. *Aerosp. Sci. Technol.*, 104(105944):105944, September 2020.

[129] Hoang Vu Phan and Hoon Cheol Park. Insect-inspired, tailless, hover-capable flapping-wing robots: Recent progress, challenges, and future directions. *Prog. Aerosp. Sci.*, 111(100573):100573, November 2019.

[130] Matěj Karásek, Florian T Muijres, Christophe De Wagter, Bart DW Remes, and Guido CHE De Croon. A tailless aerial robotic flapper reveals that flies use torque coupling in rapid banked turns. *Science*, 361(6407):1089–1094, 2018.

[131] R. Tapia, J.P. Rodríguez-Gómez, J.A. Sanchez-Diaz, F.J. Gañán, I.G. Rodríguez, J. Luna-Santamaria, J.R. Martínez-De Dios, and A. Ollero. A comparison between framed-based and event-based cameras for flapping-wing robot perception. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3025–3032, 2023.

[132] Raul Tapia, Javier Luna-Santamaria, Ivan Gutierrez Rodriguez, Juan Pablo Rodríguez-Gómez, José Ramiro Martínez-de Dios, and Anibal Ollero. Flight of the future: An experimental analysis of event-based vision for online perception onboard flapping-wing robots. *Advanced Intelligent Systems*, page 2401065, 2025.

[133] Saeed Rafee Nekoo, Ramy Rashad, Christophe De Wagter, Sawyer B Fuller, Guido de Croon, Stefano Stramigioli, and Anibal Ollero. A review on flapping-wing robots: Recent progress and challenges. *The International Journal of Robotics Research*, 2025.

[134] Martina Poletti and Michele Rucci. A compact field guide to the study of microsaccades: Challenges and functions. *Vision research*, 118:83–97, 2016.

[135] Bruce D Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI'81: 7th international joint conference on Artificial intelligence*, volume 2, pages 674–679, 1981.

[136] Diana A. Olejnik, Bardienus P. Duisterhof, Matej Karásek, Kirk Y. W. Scheper, Tom van Dijk, and Guido C. H. E. de Croon. A tailless flapping wing mav performing monocular visual servoing tasks. *Unmanned Systems*, 08(04):287–294, 2020.

[137] Yiming Wang, Qian Huang, Chuanxu Jiang, Jiwen Liu, Mingzhou Shang, and Zhuang Miao. Video stabilization: A comprehensive survey. *Neurocomputing*, 516:205–230, 2023.

[138] Marcos Roberto e Souza, Helena de Almeida Maia, and Helio Pedrini. Survey on digital video stabilization: Concepts, methods, and challenges. *ACM Computing Surveys (CSUR)*, 55(3):1–37, 2022.

[139] Wilko Guilluy, Laurent Oudre, and Azeddine Beghdadi. Video stabilization: Overview, challenges and perspectives. *Signal Processing: Image Communication*, 90:116015, 2021.

[140] Feng Liu, Michael Gleicher, Hailin Jin, and Aseem Agarwala. Content-preserving warps for 3D video stabilization. *ACM Trans. Graph.*, 28(3), July 2009.

[141] Feng Liu, Michael Gleicher, Jue Wang, Hailin Jin, and Aseem Agarwala. Subspace video stabilization. *ACM Trans. Graph.*, 30(1), February 2011.

[142] Yasuyuki Matsushita, Eyal Ofek, Xiaoou Tang, and Heung-Yeung Shum. Full-frame video stabilization. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 50–57 vol. 1, 2005.

[143] Y. Matsushita, E. Ofek, Weina Ge, Xiaoou Tang, and Heung-Yeung Shum. Full-frame video stabilization with motion inpainting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(7):1150–1163, 2006.

[144] Ken-Yi Lee, Yung-Yu Chuang, Bing-Yu Chen, and Ming Ouhyoung. Video stabilization using robust feature trajectories. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1397–1404, 2009.

[145] Yu-Shuen Wang, Feng Liu, Pu-Sheng Hsu, and Tong-Yee Lee. Spatially and temporally optimized video stabilization. *IEEE Transactions on Visualization and Computer Graphics*, 19(8):1354–1361, 2013.

[146] Zinuo You, Stamatios Georgoulis, Anpei Chen, Siyu Tang, and Dengxin Dai. GaVS: 3d-grounded video stabilization via temporally-consistent local reconstruction and rendering. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, 2025.

[147] Chao Jia, Zeina Sinno, and Brian L. Evans. Real-time 3D rotation smoothing for video stabilization. In *2014 48th Asilomar Conference on Signals, Systems and Computers*, pages 673–677, 2014.

[148] K. Sato, S. Ishizuka, A. Nikami, and M. Sato. Control techniques for optical image stabilizing system. *IEEE Transactions on Consumer Electronics*, 39(3):461–466, 1993.

[149] Fabrizio La Rosa, Maria Celvisia Virzì, Filippo Bonaccorso, and Marco Branciforte. Optical image stabilization (OIS). *STMicroelectronics. Available online: http://www. st. com/resource/en/white_paper/ois_white_paper. pdf (accessed on 12 October 2017)*, 2015.

[150] Shuaicheng Liu, Haipeng Li, Zhengning Wang, Jue Wang, Shuyuan Zhu, and Bing Zeng. DeepOIS: Gyroscope-guided deep optical image stabilizer compensation. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(5):2856–2867, 2022.

[151] Gih-Keong Lau, Yao-Wei Chin, Shih-Chun Lin, Yu-Hsiang Lai, and Boo Cheong Khoo. Spring and power in hovering ornithopters. *Advanced Intelligent Systems*, 7(7):2400477, 2025.

[152] Erzhen Pan, Xu Liang, and Wenfu Xu. Development of vision stabilizing system for a large-scale flapping-wing robotic bird. *IEEE Sensors Journal*, 20(14):8017–8028, 2020.

[153] Christophe De Wagter, Sjoerd Tijmons, Bart DW Remes, and Guido CHE de Croon. Autonomous flight of a 20-gram flapping wing mav with a 4-gram onboard stereo vision system. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4982–4987. IEEE, 2014.

[154] Juan Pablo Rodriguez-Gomez, JR Martínez-de Dios, Anibal Ollero, and Guillermo Gallego. On the benefits of visual stabilization for frame-and event-based perception. *IEEE Robotics and Automation Letters*, 2024.

[155] Adobe Premiere Pro video stabilization tutorial. `https://www.adobe.com/creativecloud/video/discover/stabilize-video.html`. Accessed: 2025-08-28.

[156] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

[157] Andy Zeng, Shuran Song, Johnny Lee, Alberto Rodriguez, and Thomas Funkhouser. TossingBot: Learning to throw arbitrary objects with residual physics. *IEEE Transactions on Robotics*, 36(4):1307–1319, 2020.

[158] Levi Burner, Cornelia Fermüller, and Yiannis Aloimonos. Embodied visuomotor representation. *npj Robot*, 3(1):30, 2025.