



UvA-DARE (Digital Academic Repository)

Recurrent motion in vision

Runia, T.F.H.

Publication date

2021

Document Version

Final published version

[Link to publication](#)

Citation for published version (APA):

Runia, T. F. H. (2021). *Recurrent motion in vision*. [Thesis, fully internal, Universiteit van Amsterdam].

General rights

It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations

If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: <https://uba.uva.nl/en/contact>, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.



Recurrent Motion in Vision

Tom Runia

Recurrent Motion in Vision

Tom Frederik Hugo Runia

This thesis was typeset by the author using \LaTeX .

Cover photo: **Zac Ong** (Courtesy of the artist, permission to reprint)

Print: Ridderprint — www.ridderprint.nl

Copyright © 2021 by Tom Runia.

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written permission of the author.

ISBN 978-94-6416-356-8

Recurrent Motion in Vision

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam
op gezag van de Rector Magnificus

prof. dr. ir. K.I.J. Maex

ten overstaan van een
door het College voor Promoties ingestelde commissie,
in het openbaar te verdedigen
op vrijdag 9 april 2021, te 15.00 uur

door

Tom Frederik Hugo Runia

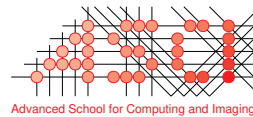
geboren te Leiden



Promotiecommissie

Promotores:	prof. dr. C.G.M. Snoek	Universiteit van Amsterdam
	prof. dr. ir. A.W.M. Smeulders	Universiteit van Amsterdam
Overige leden:	prof. dr. M. Shah	University of Central Florida
	prof. dr. T. Gevers	Universiteit van Amsterdam
	prof. dr. M. Welling	Universiteit van Amsterdam
	dr. E. Gavves	Universiteit van Amsterdam
	dr. ir. G.J. Burghouts	TNO

Faculteit der Natuurwetenschappen, Wiskunde en Informatica



The work described in this thesis has been carried out at the QUVA Deep Vision lab of the University of Amsterdam. Funding for this research was provided by Qualcomm Research. This dissertation, with identifier 416, is supported by Advanced School for Computing and Imaging (ASCI).

SUMMARY



In this thesis, *Recurrent Motion in Vision*, we explore the concept of recurrent motion patterns in video. The thesis, consisting of two parts, begins with a study on the origin of periodic motion which results in its categorization into fundamental cases. The two chapters that follow introduce novel solutions for estimating repetition in real-world videos by means of counting. In Part II we investigate recurrent motion in physical scenes for learning intrinsic object properties. Our contributions are:

Part I. Recurrent Motion in Vision

- In [Chapter 2 \(Runia et al., 2019\)](#), starting from the 3D flow field induced by a moving object, we categorize fundamental cases of intrinsic periodic motion through a decomposition of the motion. For the 2D perception of 3D periodicity as appearing in video, two viewpoint extremes are considered. What follows is the categorization of 18 fundamental cases of recurrent perception derived from the differential operators acting on the flow field.
- In [Chapter 3 \(Runia et al., 2018\)](#), we propose a novel method for repetition estimation in video that uses flow-based representations grounded in our theory. To handle cases of non-stationary repetition, we adopt the continuous wavelet transform to extract time-varying frequency information from video. For our experiments, we introduce a new video dataset for evaluating repetition counting under realistic circumstances.
- In [Chapter 4 \(Runia et al., 2019\)](#), we introduce an improved method for repetition estimation. The method focuses on spatial localization of repetitive motion directly from temporal filter responses. Furthermore, we introduce a solution to handle dynamic viewpoint changes by combining multiple flow-based representations.

Part II. Recurrent Physical Dynamics

- In [Chapter 5 \(Runia et al., 2020a\)](#), we propose an approach for measuring real-world physical properties. Our efforts concentrate

on the recurrent motion of cloth in the wind. The proposed method takes the form of an iterative refinement procedure with simulation at its core. We adopt contrastive learning and introduce a novel spectral decomposition layer to encode the cloth’s characteristics. To quantitatively evaluate our approach, we have collected real-world data by gauging the wind speed and recording flags.

- In **Chapter 6**, we consider the problem of inferring intrinsic object properties (*e.g.* mass and restitution coefficients) and learning implicit models of physical systems (*e.g.* springs and collision models). We adopt graph networks and, once more, use contrastive learning to learn physical relationships without direct supervision. Through post-training model dissection, we find evidence that our models have learned physical object properties and an implicit model of physical dynamics.



Table of Contents

Summary	vi
1 Introduction	1
Part I Recurrent Motion in Vision	13
2 Visual Periodicity	15
2.1 Introduction	15
2.2 Related Work	16
2.3 Repetitive Motion	17
2.3.1 Motion Field Decomposition	18
2.3.2 Intrinsic Periodic Motion in 3D	20
2.3.3 Visual Recurrence in 2D	22
2.3.4 Non-Static Recurrence	24
2.3.5 Non-Stationary Repetition	26
2.4 Conclusion	27
3 Repetition Estimation	29
3.1 Introduction	29
3.2 Related Work	30
3.3 Method	32
3.3.1 Flow-based Signals from Video	32
3.3.2 Continuous Wavelet Transform	34
3.3.3 Repetition Counting	36
3.3.4 Min-Cost Signal Selection	36
3.4 Datasets and Evaluation Metrics	37
3.4.1 YTSegments Dataset	37
3.4.2 QUVA Repetition Dataset	37
3.4.3 Evaluation Metrics	41
3.5 Experiments	41
3.5.1 Implementation Details	41

3.5.2	Counting Baselines	42
3.5.3	Temporal Filtering: Fourier versus Wavelets	42
3.5.4	Value of Diverse Flow Signals	44
3.5.5	Video Acceleration Sensitivity	46
3.5.6	Comparison State-of-the-Art	46
3.6	Conclusion	50
4	Improved Repetition Estimation	51
4.1	Introduction	51
4.2	Related Work	52
4.3	Method	53
4.3.1	Differential Geometric Motion Maps	53
4.3.2	Dense Temporal Filtering	57
4.3.3	Combining Spectral Power Maps	58
4.3.4	Spatial Segmentation	59
4.3.5	Repetition Counting	60
4.4	Implementation Details	60
4.4.1	Optical Flow	60
4.4.2	Motion Segmentation	61
4.4.3	Differential Geometric Motion Maps	61
4.4.4	Continuous Wavelet Transform	62
4.4.5	Repetition Counting	62
4.5	Experiments	63
4.5.1	Viewpoint Invariance	63
4.5.2	Video Acceleration Sensitivity	65
4.5.3	Motion Segmentation	66
4.5.4	Comparison to the State-of-the-Art	69
4.6	Conclusion	71
Part II	Recurrent Physical Dynamics	75
5	Cloth in the Wind	77
5.1	Introduction	77
5.2	Related Work	79
5.3	Method	81
5.3.1	Physical Similarity	82
5.3.2	Simulation Parameter Optimization	84
5.3.3	Physics, Simulation and Appearance of Cloth	84
5.3.4	Physical Model	85

5.3.5	Simulation Engine	87
5.3.6	Spectral Decomposition Network	88
5.4	Real and Simulated Datasets	92
5.4.1	Real-world Flag Videos	92
5.4.2	FlagSim Dataset	94
5.4.3	ClothSim Dataset	96
5.5	Results and Discussion	96
5.5.1	Real-world Extrinsic Wind Speed Measurement . .	96
5.5.2	Physical Similarity Quality	98
5.5.3	Real-world Intrinsic Cloth Parameter Recovery . .	98
5.5.4	Real-world Combined Parameter Refinement . . .	99
5.6	Conclusion	103
6	Learning Physical Properties and Relationships	105
6.1	Introduction	105
6.2	Related Work	107
6.3	Method	109
6.3.1	Graph Network Preliminaries	109
6.3.2	Model Architecture	111
6.3.3	Contrastive Learning	113
6.4	Experimental Setup	114
6.4.1	Physics Environments	114
6.4.2	Model and Training Details	115
6.4.3	Evaluation Metrics	115
6.5	Experiments	116
6.5.1	Correlation Analysis of Properties from Vision . .	116
6.5.2	Learning Physical Relationships from Vision . .	118
6.6	Conclusion	119
7	Conclusion	123
	Bibliography	129
	List of Publications	145
	Samenvatting – Summary in Dutch	146
	Acknowledgements	148

CHAPTER 1



INTRODUCTION

1.1 Computer Vision

The question *what does it mean to see?* has intrigued thinkers throughout humanity. David Marr ([Marr, 1982](#)) maintains that vision is first and foremost an information-processing task primarily concerned with localization and representation of objects. Accepting this notion has a fascinating consequence. *If* our vision system is merely an information-processing task, then we could reverse engineer and implement it using computer hardware given sufficient computing and memory resources. This is one of the central theses in *computer vision*.

MIT's famous summer vision internship project was a vigorous attempt to "*construct a significant part of a visual system*" ([Papert, 1966](#)). While this proved to be harder than a summer internship, a remarkable process in solving a plethora of vision tasks has been made since that summer fifty-four years ago. With this intriguing question as guidance, early work in computer vision was primarily concerned with edge detection ([Canny, 1986](#); [Otsu, 1979](#)), corner detection ([Shi and Tomasi, 1994](#)), shape representation, ([Marr and Nishihara, 1978](#)), motion and optical flow ([Lucas and Kanade, 1981](#); [Horn and Schunck, 1981](#)) and multi-scale image analysis ([Burt and Adelson, 1983](#); [Koenderink, 1984](#)). Many of these pioneering works remain to serve as the foundation of present-day computer vision. But many things have changed in the past decade.

Since the groundbreaking work of [Krizhevsky et al., 2012](#) on large-scale training of convolutional neural networks ([Fukushima, 1980](#); [LeCun et al., 1989](#)), computer vision and machine learning have become increasingly intertwined. Deep learning ([LeCun et al., 2015](#)) has revolutionized machine vision by remarkable breakthroughs in the areas of image classification ([He et al., 2016](#)), object localization ([Ren et al., 2015](#)), video action classification ([Simonyan and Zisserman, 2014](#)), im-

age synthesis (Brock et al., 2019) and unsupervised learning (Chen et al., 2020). The shared commonality between these methods is their use of high-capacity convolutional neural networks that are optimized for a particular task by learning from carefully curated visual datasets.

Despite the fast progress in modern computer vision, encoding and leveraging *motion* in video remains difficult for video understanding. On the benchmark task of action classification in video, encoding the image *appearance* is significantly more important than encoding motion information (Tran et al., 2018; Feichtenhofer et al., 2019). However, improving our ability to encode motion information is important in settings without access to color pixels such as time-of-flight cameras. Motion can also play an important role in various vision tasks such as depth estimation, spatiotemporal attention, scene navigation and structure-from-motion. In this thesis, we contribute to the understanding of motion in video. We will do so by focusing on the subset of *recurrent motion* in video.

1.2 Recurrent Motion

Recurrent motion is ubiquitous in the visual world around us (Figure 1.1). In a typical mundane scene such as having breakfast, visual rhythm can be perceived as the stirring in our coffee, the cutting of oranges, spreading marmalade on toast and the chewing motion of our jaws. Once acquainted with the awareness of recurrent motion, its abundance in the visual world becomes evident. It appears in a wide array of human activities such as sports, music-making and cooking. In natural scenes, we confront it as leaves in the wind, waves in the sea or the drumming of a woodpecker. Furthermore, encounters of visual repetition in urban environments are everywhere, whether flashing lights, cars on a highway, spinning of wind turbines or the waving of a pedestrian.

Given the ubiquity of recurrent motion in the visual world, it is worthwhile to understand its origin and how it is perceived by our visual system. There is substantial evidence that the human vision system leverages repetitive motion as one of its strongest visual cues for recognizing actions, warning for danger and guiding our attention (Repp and Penel, 2004; Sejdić et al., 2012; H. Li et al., 2014; Brandon and Saffran, 2011). Its cognitive importance is furthermore underlined by the early development of infants’ ability to recognize visual rhythm.



Figure 1.1. Recurrent motion is pervasive in the visual world around us. We perceive it as human movement, the bouncing of a ball, the curling of a flag in the wind, and as rolling waves in the sea. In this thesis, we will explore the origins, appearance and practical value of recurrent motion in vision.

Specifically, it was shown that 7-month old infants are not capable to distinguish between stationary visual rhythms, whereas they are able to discriminate visual rhythm appearing as repetitive *motion* (Brandon and Saffran, 2011).

Our strong ability to process recurrent motion has given rise numerous applications in everyday scenes. The beacon lights mounted upon emergency vehicles provide an important signal for visual warning (Bisley, 2011). Movie directors have used temporal modulation in the form of blinking lights as a powerful tool for capturing our attention and intensifier of dramatic effects (Truong and Venkatesh, 2001). And, strongly periodic recurrent motion such as a metronome can be utilized for measuring time and temporal synchronization (Hove et al., 2013).

One of the primary characteristics of recurrent motion is its intrinsic time-scale. At one end of the human-observable temporal spectrum, we can find the high-frequency vibrating of a string, while on the opposite end of the spectrum we can recognize day-night cycles or even the annual change of seasons. Different time-scales may serve a distinct cognitive purpose, for example flashing lights are important for spatial visual attention while the recurring daily sunrise yields a strong sense of time. Understanding these distinct uses and understanding the temporal resolution and cognitive selectiveness to recurrent stimuli is important for utilizing its effectiveness in real-world scenes such as an early visual warning while driving at high-speed on a motorway.

Apart from cognitive importance and academic curiosity, understanding the origins of recurrent motion in vision and our ability to detect, localize and count it has numerous important applications. In computer vision, the presence of recurrent motion has been leveraged to infer 3D structure (Belongie and Wills, 2006), estimate depth (Huang et al., 2016), classify human actions (Lu and Ferrier, 2004), categorize sports video (Johansson, 1973), calibrate cameras (Huang et al., 2016) and to find duplicate video content (Covell et al., 2006). These examples underscore the value of recurrent motion in real-world problems and, therefore, serve as the primary motivation for our study on the origin, appearance and value of recurrent motion in vision.

Due to its importance for visual understanding, estimating recurrent motion from video has received increasing attention in recent years. To handle a wide spectrum of temporal scales, early work predominantly relied on signal processing methods such as Fourier transforms (Polana and Nelson, 1997), time-series autocorrelation (Cutler and Davis, 2000) or tunable time-frequency filters (Burghouts and Geusebroek, 2006). More recently, learning-based methods utilizing convolutional neural networks for learning spatial representations with temporal sampling approaches have become the better-performing approaches on more challenging real-world datasets containing a high-variety of visual scenes (Levy and Wolf, 2015; Dwibedi et al., 2020; H. Zhang et al., 2020; Karvounas et al., 2019). Throughout this thesis, we consider more traditional signal processing as well as deep learning and hybrid methods for estimating recurrent motion in video.

1.3 Research Questions

This thesis takes a broad view of the concept of recurrent motion in video. Our story is structured in two parts: Part I explores the origin, appearance and *counting* of recurrent motion in video. Part II focuses on the manifestation and value of recurrent motion in *physical* environments. We proceed by formulating our research questions that will serve as guidance throughout the thesis.

Part I. Recurrent Motion in Vision

Motivated by our practical desire to estimate repetitions in real-world video, our first question focuses on delivering a comprehensive categorization of 3D periodic motion as it exists in our visual world. We are also interested in the appearance of the recurrent motion on the image plane as captured on video. This brings us to our first research question:

Research Question 1: What periodic motion types exist for objects moving in 3D space and what are their appearance types on the 2D image plane?

In [Chapter 2](#) (Runia et al., 2019), we propose a new categorization of periodic motion in video as induced by an object moving through 3D space. To understand the origin and appearance of visual repetition we rethink the theory of periodic motion inspired by existing work. We follow a *differential approach* by starting from the divergence, gradient and curl components of the 3D flow field. From the decomposition of the motion field and its temporal dynamics, we derive three motion types and three motion continuities to arrive at 3×3 fundamental cases of intrinsic periodicity in 3D. For the 2D perception of 3D intrinsic periodicity, the observer’s viewpoint can be somewhere in the continuous range between two viewpoint extremes. This insight leads us to 18 fundamental cases for the 2D perception of 3D intrinsic periodic motion. We further underscore the existence of *non-stationary* and *non-static* periodic motion in video. Non-stationarity refers to weakly-periodic motion such as the pedal motion of an accelerating cyclist. The non-static appearance of recurrent motion refers to a change in the perceived

motion field, for example by viewpoint change or a gradual change in observed motion.

After the theory-focused exploration on the origin and appearance of recurrent motion, we shift our attention to estimating repetition in video. This leads us to our second research question:

Research Question 2: How can we detect, localize and count repetitive motion in arbitrary realistic video?

We attempt to answer this broad question by proposing two methods for repetition estimation in [Chapter 3](#) (Runia et al., 2018) and [Chapter 4](#) (Runia et al., 2019). Specifically, we focus on the problem of *counting repetition* in video. Existing work shows good results under the assumption of static and stationary periodicity. However, as realistic video is rarely perfectly static and stationary, the often preferred Fourier-based measurements is inapt. Instead, both our methods adopt the *continuous wavelet transform* to better handle non-static and non-stationary video dynamics. In practice, to deal with the variety of repetitive appearances, our theory implies measuring time-varying flow and its differentials (divergence, gradient and curl) over segmented foreground motion. To handle the non-static appearance of recurrent motion, our initial method ([Chapter 3](#)) relies on an existing method for foreground motion segmentation. In [Chapter 4](#), we concentrate on the spatial localization aspect of repetitive motion by segmenting each video frame from the wavelet spectrum directly. This eliminates the need for a decoupled localization method and even improves counting performance. For experiments in both chapters, we introduce a new dataset, better-reflecting reality by including non-static and non-stationary videos. Our video dataset contains a wide variety of commonplace scenes such as playing tennis, performing push-ups, human activities and natural scenes. On the task of counting repetitions, our methods compare favorably to recent learning-based approaches.

Part II. Recurrent Physical Dynamics

In the second part of this thesis, we transition to the study of recurrent motion in *physical environments*. This is motivated by our hypothesis that visual recurrence appears in a broad range of physical phenomena and

can be utilized for learning from visual observations. The exploration starts with our third research question:

Research Question 3: Can we use recurrent motion in video for real-world physical measurements?

In [Chapter 5 \(Runia et al., 2020a\)](#), we explore the possibility of measuring real-world physical quantities from a video of cloth in the wind. Our main contribution is a method for measuring latent physical properties from the recurrent motion of cloth in the wind. As obtaining real-world measurement data and corresponding visual observations can be difficult or expensive, our method uses *physics simulations* during training. Our algorithmic solution is an iterative refinement procedure with simulation at its core. Training takes the form of *contrastive learning* on a dataset of simulated cloth video clips. During inference, we measure real-world physical parameters by gradually refining simulation parameters to increase physical similarity with its real-world observation. The physical correspondence is measured using an embedding function that maps physically similar examples to nearby points. We consider a case study of cloth in the wind, with curling *flags* as our leading example; a seemingly simple phenomenon but physically highly involved. Based on the physics of cloth and its visual manifestation, we propose a particular instantiation of the embedding function. For this mapping, modeled as a deep network, we introduce a new *spectral decomposition layer* that decomposes a video volume into its temporal spectral power and corresponding frequencies. Our experiments demonstrate that the proposed method compares favorably to prior work on the task of measuring cloth material properties and external wind force from a real-world video.

The fourth and final research question recognizes a more abstract notion of recurrent motion. Given the repeated visual observation of a particular physical interaction, can we learn to infer physical properties or relationships? We formalize this in the following research question:

Research Question 4: What do recurrent object interactions reveal about physical object properties and dynamics?

Chapter 6 (Runia et al., 2020b) explores this question by focusing on simple object-centric physics environments for which we can conveniently generate novel episodes. Specifically, we consider the problem of inferring intrinsic object properties (*e.g.* mass and coefficients of restitution) and learning implicit models of physical systems (*e.g.* springs and collision models). Unlike existing work, our emphasis is on learning these aspects from repeated visual observations only, rather than having access to the underlying physical state of the environment. Crucially, this requires the decomposition of a visual observation into individual object representations. Our solution uses three models that operate in tandem. The visual encoder disentangles the observations into an abstract *factorized state representation*. Based on a sequence of abstract object states, the property predictor, implemented as *graph network*, encodes physically relevant object properties. Finally, the dynamics predictor, another graph network, is conditioned on the object properties and the current observation to predict the system’s future state. Without having access to true object states, and to avoid placing a loss in pixel space, we employ *contrastive learning* to jointly train the three networks on three new physics simulation datasets. Through post-training model dissection using correlation analysis, we find evidence that our models have learned both physical object properties and an implicit model of physical dynamics.

1.4 Co-authorship and Roles

For each chapter of this thesis we here declare the authors' contributions:

Chapter 2.

Tom F.H. Runia, Cees G.M. Snoek, and Arnold W.M. Smeulders (2018). "*Real-World Repetition Estimation by Div, Grad and Curl*". In: Conference on Computer Vision and Pattern Recognition (CVPR). (Runia et al., 2018)

- T.F.H. Runia All aspects
- C.G.M. Snoek Insight and supervision
- A.W.M. Smeulders Insight and supervision

Chapter 3.

Tom F.H. Runia, Cees G.M. Snoek, and Arnold W.M. Smeulders (2018). "*Real-World Repetition Estimation by Div, Grad and Curl*". In: Conference on Computer Vision and Pattern Recognition (CVPR). (Runia et al., 2018)

- T.F.H. Runia All aspects
- C.G.M. Snoek Insight and supervision
- A.W.M. Smeulders Insight and supervision

Chapter 4.

Tom F.H. Runia, Cees G.M. Snoek, and Arnold W. M. Smeulders (2019). "Repetition Estimation". In: International Journal of Computer Vision (IJCV). (Runia et al., 2019)

- T.F.H. Runia All aspects
- C.G.M. Snoek Insight and supervision
- A.W.M. Smeulders Insight and supervision

Chapter 5.

Tom F.H. Runia, Kirill Gavriluk, Cees G.M. Snoek, and Arnold W.M. Smeulders (2020). “Cloth in the Wind: A Case Study of Estimating Physical Measurement through Simulation”. In: Conference on Computer Vision and Pattern Recognition (CVPR). (Runia et al., 2020a)

- T.F.H. Runia All aspects
- K. Gavriluk Help with experiments of Section 5.5.3
- C.G.M. Snoek Insight and supervision
- A.W.M. Smeulders Insight and supervision

Chapter 6.

Tom F.H. Runia, Cees G.M. Snoek, and Arnold W. M. Smeulders (2020). “Learning Physical Properties and Relationships from Visual Observations”.

- T.F.H. Runia All aspects
- C.G.M. Snoek Insight and supervision
- A.W.M. Smeulders Insight and supervision



We proceed with the main body of work in which we focus on the research questions through their dedicated chapters. Chapter 7 revisits the research questions by individually answering them, addressing the limitations of our propositions and suggesting future research.

PART I

RECURRENT MOTION IN VISION

“Judge a man by his questions rather than by his answers.”

— Voltaire

PART 1 of this thesis explores the concept of visual repetition in video. The first chapter introduces new theory on the categorization of visual periodicity as it would appear in video. Building upon the theoretical groundwork, in Chapter 3 and Chapter 4 we present two methods for estimating repetition in real-world videos. In particular, we will focus on the task of *counting* repetitions as its evaluation remains meaningful in the absence of strongly periodic motion.

CHAPTER 2



VISUAL PERIODICITY

2.1 Introduction

Visual repetitive motion is common in our everyday experience as it appears in sports, music-making, cooking and other daily activities. In natural scenes, it appears as leaves in the wind, waves in the sea or the drumming of a woodpecker, whereas our encounters of visual repetition in urban environments include blinking lights, the spinning of wind turbines or a waving pedestrian. In this chapter, we reconsider the theory of periodic motion starting from a differential analysis of the flow field.

Understanding the categorization of repetitive motion and improving our ability to estimate repetition in realistic video is important in numerous aspects. In computer vision, periodic motion has proven to be useful for action classification (Goldenberg et al., 2005; Lu and Ferrier, 2004), action localization (Laptev et al., 2005; Sarel and Irani, 2005), human motion analysis (Albu et al., 2008; Ran et al., 2007), structure from motion (Belongie and Wills, 2006; X. Li et al., 2018), animal behavior study (Davis et al., 2000) and camera calibration (Huang et al., 2016). From a biological perspective, repetition is fascinating as the human visual system relies on rhythm and periodicity to approximate velocity, estimate progress and to trigger attention (Johansson, 1973).

In this chapter, to understand the origin and appearance of visual repetition we rethink the theory of periodic motion inspired by existing work of Pogalin et al., 2008 and Davis et al., 2000. We follow a differential geometric approach, starting from the divergence, gradient and curl components of the 3D flow field. From the decomposition of the motion

This chapter is based on our IJCV 2019 (Runia et al., 2019) and CVPR 2018 (Runia et al., 2018) publications.

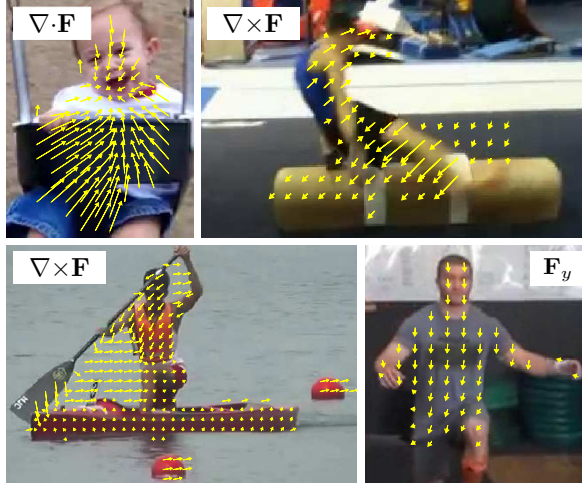


Figure 2.1. There is great diversity in appearance of repetitive motion. We decompose the motion field into its fundamental components. Here we visualize the motion fields as optical flow arrows over the foreground motion with the visually dominant motion field component indicated in the white box.

field and its temporal dynamics, we derive three *motion types* and three *motion continuities* to arrive at 3×3 fundamental cases of intrinsic periodicity in 3D. Next, we consider the appearance of the periodicity on the 2D image plane. For the 2D perception of 3D intrinsic periodicity, the observer’s viewpoint can be somewhere in the continuous range between two viewpoint extremes. Finally, we arrive at 18 fundamental cases for the 2D perception of 3D periodic motion.

2.2 Related Work

In real-world video, periodic motion emerges in a wide variety of appearances (see [Figure 2.1](#)). We reconsider the theory of periodic motion by proposing a classification of fundamental periodic motion types starting from the 3D motion field tied to a moving object. Using first-order differential analysis, we decompose the motion field into its primitive components. The work of [Koenderink and Doorn, 1975](#) delivered inspiration for our theoretical derivation of repetitive motion types from the flow field. Furthermore, our derivation shares similarity with the Helmholtz-Hodge eigenvalue decomposition ([Abraham et al., 1988](#)) of

the flow field’s Jacobian matrix, which finds its use mainly in flow field topology for fluid dynamics and electrodynamics. Although there is resemblance to the differential decomposition of the motion field, we reach a novel classification of periodic motion patterns. In the two subsequent chapters, we will use these insights for establishing methods for repetition estimation in video.

In the context of periodic motion, existing works (Davis et al., 2000; Pogalin et al., 2008) have proposed categorizations of motion patterns. Davis et al., 2000 consider a simple sinusoidal model to characterize periodic motion and link each type to animal behavior. In terms of periodic motion categorization, our exploration bears resemblance to the study of Pogalin et al., 2008. The authors identify four visually periodic motion types (translation, rotation, deformation and intensity variation) complemented with three cases of motion continuity (oscillating, constant and intermittent) in the field of view. In this chapter, we take a more principled approach starting from the 3D motion field. Specifically, we show that a fundamental categorization of periodic motion types emerge from the decomposition of the flow field and the motion direction over time. Building on this, the projection of 3D periodicity on a 2D image has to take into account the continuous nature of the viewpoint which we address explicitly in theory and experiments.

Although not directly related to our work, first-order differential geometric motion representations have been used extensively as spatiotemporal video descriptors. For example, Klaser et al., 2008 propose a spatial multi-scale motion descriptor based on first-order differential motion and uses integral videos for efficient computation. Along similar lines, MoSIFT (Chen and Hauptmann, 2009) uses spatial interest points and enforces sufficient temporal dynamics to eliminate candidate points. In terms of motion descriptors, our work bears resemblance to the Divergence-Curl-Shear descriptor proposed by Jain et al., 2013. Their strong empirical results on the task of action classification using differential-based descriptors are in agreement with our findings for repetition estimation.

2.3 Repetitive Motion

Visual repetition is defined as a recurring pattern over space or time in the 3D world. In this chapter, we focus on temporally repetitive motion

rather than spatially repetitive patterns such as a texture. Consequently, we believe that the 3D motion field induced by a moving object is the right starting point for our theoretical analysis.

We consider a moving object and observer positioned in a 3D world specified by the Cartesian coordinates $\mathbf{x} = (x_1, x_2, x_3)$ at time t . Formally, intrinsic periodic motion is defined as the reappearance of the same 3D flow $\mathcal{F}(\mathbf{x}, t)$ induced by the motion of an object over time.

$$\mathcal{F}(\mathbf{x}, t) = \mathcal{F}(\mathbf{x} + \mathbf{S}, t + T). \quad (2.1)$$

In which the parameter T denotes the period over time and \mathbf{S} corresponds to a period over space. We initially exclude the trivial case of a constant flow field inducing periodic appearance due to a reappearing texture on the object's surface. Starting from the motion field, we follow a differential approach to decompose the field into its elementary components. Ultimately, we arrive at nine fundamental cases of intrinsic periodic motion in 3D.

2.3.1 Motion Field Decomposition

In 3D Cartesian space, the gradient of the flow $\nabla \mathcal{F}(\mathbf{x}, t)$ is described by the Jacobian matrix $\mathbf{J} \in \mathbb{R}^{3 \times 3}$ containing all first-order partial derivatives of the vector field. Specifically, the Jacobian is defined as:

$$(\nabla \mathcal{F})_{ij} = \frac{\partial \mathcal{F}_i}{\partial x_j}, \quad (2.2)$$

where i and j are dimension indices and we omit the position \mathbf{x} and time t for brevity. From the first-order partial derivatives contained in the Jacobian, three fundamental components of the motion field can be recognized (Abraham et al., 1988). Specifically, the Jacobian \mathbf{J} can be decomposed into a sum of a diagonal part \mathbf{D} , a symmetric part \mathbf{E} and an anti-symmetric part \mathbf{R} such that:

$$\nabla \mathcal{F} = \mathbf{D} + \mathbf{R} + \mathbf{E}. \quad (2.3)$$

This is similar to the Helmholtz-Hodge vector field decomposition, which distinguishes divergence-free and curl-free components of a motion field and is well-known in fluid dynamics. Firstly, the diagonal part of the Jacobian \mathbf{J} takes the form of:

$$\mathbf{D} = \text{diag} \left(\frac{\partial \mathcal{F}_1}{\partial x_1}, \frac{\partial \mathcal{F}_2}{\partial x_2}, \frac{\partial \mathcal{F}_3}{\partial x_3} \right). \quad (2.4)$$

The trace of this matrix defines the *divergence* of the field:

$$\nabla \cdot \mathcal{F} = \text{trace}(\mathbf{D}). \quad (2.5)$$

In other words, the divergence is a scalar field representing the amount of outward flux from an infinitesimal volume around a given point.

Secondly, the anti-symmetric part \mathbf{R} , referred to as the spin- or rotation matrix, is given by:

$$\mathbf{R} = \frac{1}{2}(\mathbf{J} - \mathbf{J}^T), \quad (2.6)$$

with its elements defined as:

$$\mathbf{R}_{ij} = \frac{1}{2} \left(\frac{\partial \mathcal{F}_i}{\partial x_j} - \frac{\partial \mathcal{F}_j}{\partial x_i} \right). \quad (2.7)$$

From the elements of the spin matrix we can recognize the *curl* of the flow field. More specifically, the curl of the 3D flow field is defined as:

$$\nabla \times \mathcal{F} = \left[\frac{\partial \mathcal{F}_3}{\partial x_2} - \frac{\partial \mathcal{F}_2}{\partial x_3}, \frac{\partial \mathcal{F}_1}{\partial x_3} - \frac{\partial \mathcal{F}_3}{\partial x_1}, \frac{\partial \mathcal{F}_2}{\partial x_1} - \frac{\partial \mathcal{F}_1}{\partial x_2} \right]^T. \quad (2.8)$$

This vector field describes the rotation around a given point.

Finally, the last component is given by the symmetric part:

$$\mathbf{E} = \frac{1}{2}(\mathbf{J} + \mathbf{J}^T) \quad (2.9)$$

with its elements given by:

$$\mathbf{E}_{ij} = \frac{1}{2} \left(\frac{\partial \mathcal{F}_i}{\partial x_j} + \frac{\partial \mathcal{F}_j}{\partial x_i} \right). \quad (2.10)$$

This trace-free matrix is known as the deformation tensor and associated with the *shear* of the flow field. In [Figure 2.2](#) we illustrate three motion fields with either pure divergent, rotational or shear flow.

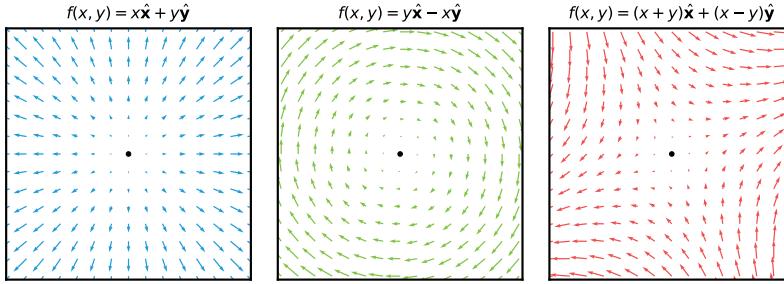


Figure 2.2. Three 2D flow fields with fundamentally different characteristics that emerge from the decomposition of the motion Jacobian \mathbf{J} . *Left:* Pure divergent flow field with outward flux often associated with expansion or depth perception. *Center:* Pure rotational flow field also referred to as vorticity or curl. *Right:* Flow field with a pure shear component related to the deformation tensor. The shear component is divergence- and curl-free as the opposing terms cancel out. In real-world video, shear is generally negligible compared to divergence and curl.

2.3.2 Intrinsic Periodic Motion in 3D

Based on the decomposition of the flow field, we here introduce the categorization of intrinsic periodic motion in 3D.

Motion Types

For an object moving periodically through the 3D space, the decomposition of the flow field tied to the object can be used to characterize the type of motion. A non-rigid object that is expanding or contracting along one or more axes will produce a purely divergent flow field $\nabla \cdot \mathcal{F}$. Examples include: inflating a balloon or a pulsing anemone. Moreover, a flow field exclusively containing curl $\nabla \times \mathbf{F}$ emerges with rotational motion such as a spinning wheel or tightening a bolt. Finally, shear is associated with deformation or stress on a surface caused by opposing forces parallel to the cross-section of a body. Shear predominantly plays a role for materials with high-elasticity (*e.g.* fluids) or in the presence of large forces (*e.g.* solid mechanics). Under normal everyday circumstances, the 3D motion field's shear component is negligible as excessive forces are required to deform the material. For softer materials such as foam, paper or plastics, the shear component can be measurable but this is rare in practice. Based on its rare appearance in common real-world scenarios, we therefore omit shear from the remainder of this chapter.

Using the spatial divergence and curl operators we can derive three basic 3D motion types. The three motion types that emerge are:

$$\begin{aligned}
\text{translation: } \nabla \times \mathcal{F}(\mathbf{x}, t) &= \mathbf{0}, \quad \nabla \cdot \mathcal{F}(\mathbf{x}, t) = 0 \\
\text{rotation: } \nabla \times \mathcal{F}(\mathbf{x}, t) &\neq \mathbf{0}, \quad \nabla \cdot \mathcal{F}(\mathbf{x}, t) = 0 \\
\text{expansion: } \nabla \times \mathcal{F}(\mathbf{x}, t) &= \mathbf{0}, \quad \nabla \cdot \mathcal{F}(\mathbf{x}, t) \neq 0.
\end{aligned} \tag{2.11}$$

These motion types are tied to a particular 3D motion field of pure form. In practice there may be a mixture of types. In [Chapter 3](#), we are aiming to estimate repetition from real-world video. Consequently, our method leverages first-order differential motion maps and determines which yields the strongest response in the object's motion.

Motion Continuities

By its nature, periodic motion contains a temporal component, we here transition to the temporal dynamics of the time-varying motion field. Consecutive measurements of the flow field $\mathcal{F}(\mathbf{x}, t)$ produce a time-varying motion field with particular temporal dynamics. Depending on the type of motion, the motion field needs to satisfy one of the following necessary periodic conditions:

$$\begin{aligned}
\nabla \mathcal{F}(\mathbf{x}, t) &= \nabla \mathcal{F}(\mathbf{x} + \epsilon, t + T) \\
\nabla \times \mathcal{F}(\mathbf{x}, t) &= \nabla \times \mathcal{F}(\mathbf{x} + \epsilon, t + T) \\
\nabla \cdot \mathcal{F}(\mathbf{x}, t) &= \nabla \cdot \mathcal{F}(\mathbf{x} + \epsilon, t + T),
\end{aligned} \tag{2.12}$$

where ϵ denotes a translation as the object's periodicity may be superposed on translation. For robustness, our method presented in [Chapter 3](#) will measure both $\mathcal{F}(\mathbf{x}, t)$ and $\nabla \mathcal{F}(\mathbf{x}, t)$. From the direction and temporal dynamics of motion, three distinct periodic motion continuities can be distinguished: *constant*, *intermittent* and *oscillating* periodicity. As with the motion types, in practice, the motion continuity may be a mixture between types. For both intermittent and oscillating motion, repetitive nature is intrinsically in the temporal dynamics whereas for constant motion to appear repetitively, there will be special conditions on the object's texture or albedo to be periodic.

Categorization of Periodic Motion

The intrinsic periodicity in 3D does not cover all perceived recurrence in an image sequence. For the trivial cases of constant translation and constant expansion in 3D, the perceived recurrence will appear when a repetitive chain of objects (*e.g.* a conveyor) or a repetitive appearance (*e.g.* texture on a car tire) on the object is aligned with the motion. In such cases, the recurrence will also be observed in the field of view. For constant rotation, the restriction is that the appearance cannot be constant over the surface, as no motion, let alone recurrent motion would be observed. In the rotational case, any rotational symmetry in appearance will induce a higher-order recurrence as a multiplication of the symmetry and the rotational speed.

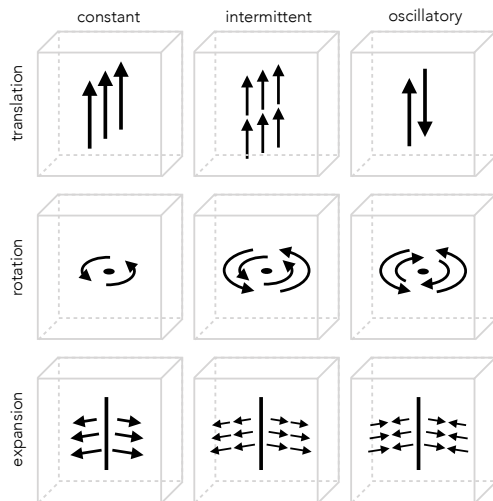
For the our interest in periodic motion, the basic three *motion types* and three *motion continuities* organize in a 3×3 Cartesian table of nine fundamental periodic motion types. The flow abstractions and corresponding examples of these cases are visualized in [Figure 2.3](#). This is the list of fundamental cases, whereas mixtures will often appear in the real-world. In practice, some cases are ubiquitous, while for others it is hard to find examples at all.

2.3.3 Visual Recurrence in 2D

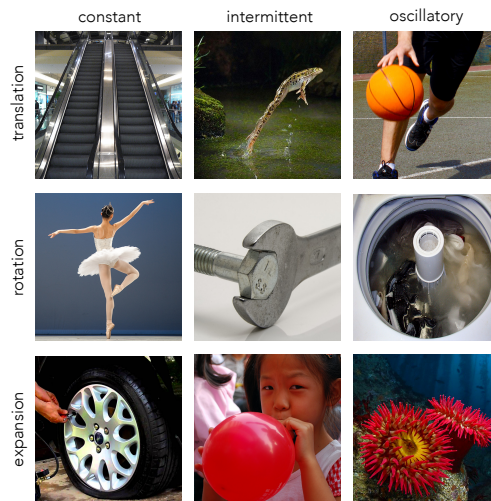
Up until now, we have considered the intrinsic periodicity in 3D. We reserve the term *recurrence* for the 2D observation of the 3D periodicity. Recurrence in the field of view is defined by:

$$\mathbf{F}(\mathbf{x}', t) = \mathbf{F}(\mathbf{x}' + \epsilon', t + T) \quad (2.13)$$

where $\mathbf{F}(\mathbf{x}', t)$ is the perceived flow in 2D image coordinates \mathbf{x}' . The observed displacement is denoted by ϵ' and T is the temporal period. The underlying principle is that *the same period length T will be observed in both 3D and 2D for all cases of periodicity*. This permits us to measure 3D motion periodicity T from the 2D flow field. Only in some extraordinary cases, the periodicity may change due to a partial or complete occlusion, or the periodic motion disappears entirely due to lack of texture or albedo from a given viewpoint (*e.g.* a constantly rotating textureless disk). However, these are exceptional cases as the general principle applies that the temporal period is viewpoint invariant.



(a) Flow Abstractions in 3D



(b) Examples in Real Life

Figure 2.3. 3×3 Cartesian table of the *motion type* times the *motion continuity*. These are the basic cases of periodicity in 3D emerging from the motion field decomposition and the temporal dynamics. The examples are: escalator, leaping frog, bouncing ball, pirouette, tightening a bolt, laundry machine, inflating a tire with repetitive texture, inflating a balloon and a breathing anemone.

The camera position relative to the object's motion is crucial for the perception of the flow field on the camera plane. There are two fundamentally different viewpoints: the *frontal* view and the *side* view:

frontal view: on the main axis of motion

side view: perpendicular to the main axis of motion.

For translation, there is one main axis and two perpendicular axes, which are both identical for our purpose. There is no distinction between the two perpendicular views as their perception is equivalent. Similarly, for rotation, the two perpendicular cases are also indistinguishable. For expansion there are one, two or three axes of expansion, again leaving us with the frontal case and the perpendicular case as the two fundamental cases. Consequently, for all cases considered, a distinction between frontal view and side view suffices. As a result, the perceived recurrence is defined on the continuous range between the two extreme viewpoints. Combining the two viewpoint extremes with the nine cases of periodic motion we arrive at the classification of 18 basic cases of *perceived visual recurrence*. An abstraction of the 2D field for the 18 cases is visualized in [Figure 2.4](#). The two views are the end of a continuous range of viewpoints. Generally, an actual viewpoint will be somewhere in between the frontal view and the side view. This leaves the flow field asymmetrical or skewed, either in divergence, gradient or curl. Nonetheless, as long as period length T can be measured from the observed signal, the skewed or asymmetric observation will not affect the recurrent nature nor the period of the 3D motion field.

2.3.4 Non-Static Recurrence

Relative motion between the moving object and the observer adds another level of complexity. In particular with recurrent motion: (1) the camera can move because the camera is mounted on the moving object itself; or (2) the camera is following the target of interest; or (3) the camera is in motion independent of the motion of the object. For the first two cases, the camera motion reflects the periodic dynamics of the object's motion. The flow field may be outside the object, but otherwise it displays a complementary pattern in the flow field.

In the first case, the periodically moving camera will produce a global repetitive flow field as opposed to local repetitive flow when the object

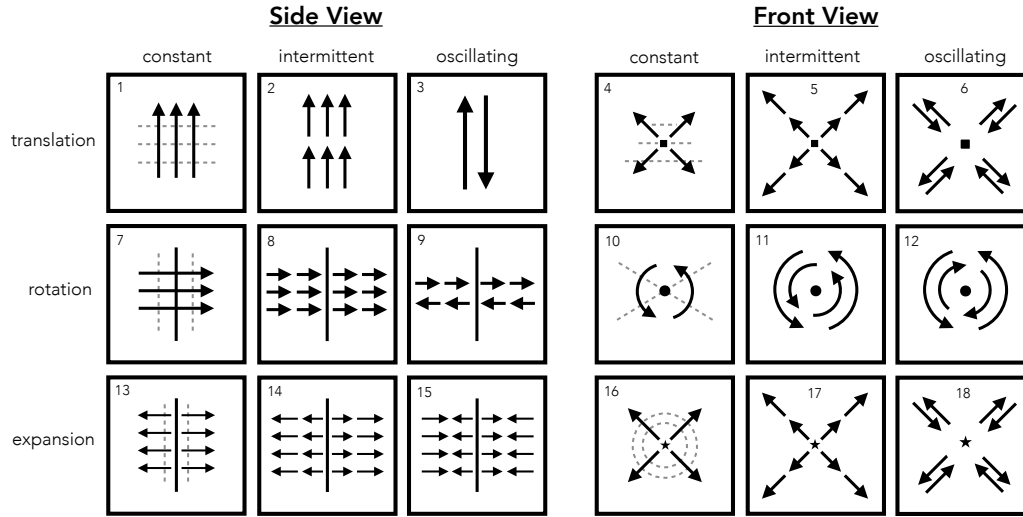


Figure 2.4. Observed flow in a 2D image sequence: the 18 fundamental cases for 2D perception of 3D recurrence. The perception follows from the motion pattern (3×), motion continuity (3×) and the viewpoint on the continuous interval between the two extremes: side and front view. ↑ denotes flow direction, ■ denotes a vanishing point, ● denotes a rotation point, ★ denotes expansion point. Dashed grey lines for constant motion indicate the need for texture to perceive recurrence. Pairs 4-16, 5-17 and 6-18 appear similar at first but vary in temporal signal profile.

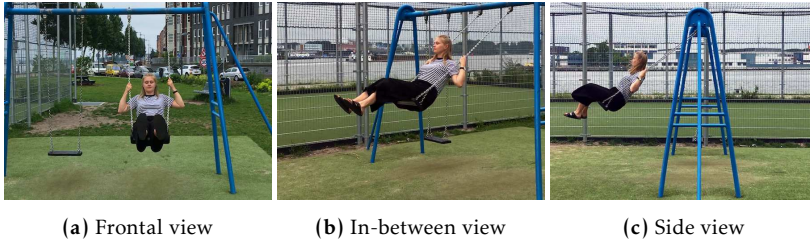


Figure 2.5. Example video displaying *girl on a swing* captured from three distinct viewpoints. Moving from one end of the continuous viewpoint spectrum (frontal) to the other (side) results in a dramatic change of motion appearance. The in-between viewpoint leaves the motion measurements either skewed or asymmetrical. In [Chapter 4](#), we combine multiple differential motion representations to handle such camera transitions.

itself is moving. In particular, the third case demands the removal of the camera motion before the repetitive motion analysis. This situation occurs frequently in real-world scenarios. Therefore, particular attention needs to be paid to camera motion independent of the target’s motion. When the viewpoint changes from frontal to side view due to camera motion, the analysis will be inevitably hard. [Figure 2.5](#) illustrates the dramatic changes in the flow field when the camera changes from one extreme viewpoint (side) to the other (frontal), or vice versa. Our first method for repetition estimation ([Chapter 3](#)) is unable to handle such variations as it selects the single most discriminative flow representation. However, we improve upon this with our method outlined in [Chapter 4](#) by handling appearance changes by simultaneously using multiple motion representations and combining their temporal filter responses.

2.3.5 Non-Stationary Repetition

A recurrent signal is said to be stationary when the period length is constant over time. In the initial steps of periodicity analysis, it was assumed the periodic signal was near-stationary. However, decay in frequency or acceleration is common in realistic video. An example of a rower accelerating as plotted in the time-frequency spectrum is displayed in [Figure 2.6](#). In [Chapter 3](#), we will observe that non-stationary often appears in real-world video. Therefore, in contrast to existing work ([Pogalin et al., 2008](#); [Levy and Wolf, 2015](#)), we will loosen the stationarity assumption leaving the option of acceleration open. More precisely the

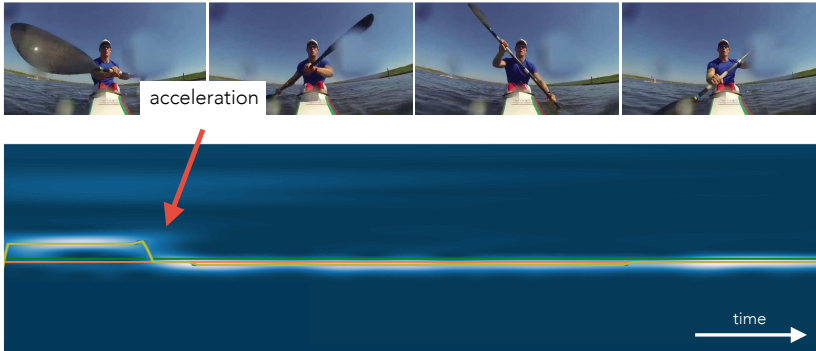


Figure 2.6. Non-stationary motion often appears in real-world video. This example shows a rower accelerating as plotted in the time-frequency space. The vertical axis of the spectrum denotes the wavelet scale, inversely proportional to the frequency. The sudden acceleration appears as shift of the maximum power in time-frequency space.

methods presented in the next two chapters will use the continuous wavelet transform for spectral decomposition of the video.

Furthermore, even when object motion and camera are both static, for none of the three intrinsic motion types (translation, rotation, expansion), a point on the object will be at the same position in the camera field all the time. Under the double static condition, a point will just return to the same point on the camera field. As the intermediate points on the object or background have an arbitrary albedo and radiate an arbitrary luminance, it will not produce a sinusoidal signal in general. This is noteworthy as previous work (Cutler and Davis, 2000; Liu and Picard, 1998; Pogalin et al., 2008) implicitly assumes such a signal by considering the Fourier transform or variants.

2.4 Conclusion

This chapter has explored the categorization of repetitive motion into its fundamental cases. Starting from the 3D motion induced by an object moving through space, we derive primitive cases of motion by differential decomposition of the flow field. From the decomposition of the motion field and its temporal dynamics, we have identified three motion types and three motion continuities to arrive at 3×3 cases of intrinsic periodicity in 3D. Next, we have considered the appearance of these

cases on the 2D image plane when perceived by a video camera. The observer's viewpoint can be somewhere in the continuous range between two viewpoint extremes. Consequently, we arrive at 18 fundamental cases for the 2D perception of 3D intrinsic periodic motion. In the two chapters that follow, we will use the flow field differentials to encode motion in video for our repetition estimation methods.

CHAPTER 3



REPETITION ESTIMATION

3.1 Introduction

Building upon the theory of visual periodicity, we proceed by considering the tangible problem of estimating repetition in video. As we have seen, repetitive motion in real-world video is ubiquitous. Examples of its appearance include performing push-ups, cutting a melon or playing violin (Figure 3.1). Existing methods for counting repetitions show good results under the assumption of static and stationary periodicity (Levy and Wolf, 2015; Pogalin et al., 2008). As realistic repetitive motion is rarely perfectly static and stationary, the often preferred Fourier-based measurements are inapt. In short, existing work focuses on video that is static in every aspect of repetition. As real life is more complex, the method presented in this chapter relies on motion foreground segmentation to localize the salient motion and handle non-static video. Furthermore, our empirical findings demonstrate that fixed-period Fourier analysis (Cutler and Davis, 2000; Pogalin et al., 2008; Polana and Nelson, 1997) to be unsuitable for repetition estimation in real-world video as non-stationarity often appears. To permit non-stationary video dynamics, we adopt the wavelet transform for decomposing video signals into a time-frequency spectrum.

The contributions of this chapter are the following. Building upon the theory of visual repetition (Chapter 2), we propose a method for counting repetitive actions in video. To estimate repetition in video under realistic circumstances, we compute a diverse flow-based differential representation over the motion foreground segmentation as suggested by the theoretical groundwork. Our method uses wavelets to handle

This chapter is based on our CVPR 2018 publication (Runia et al., 2018).

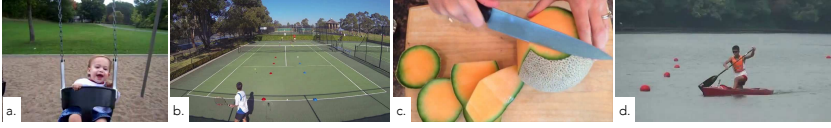


Figure 3.1. Examples from our *QUVA Repetition* dataset, containing videos with repetitive motion such as sports, cooking, music-making, and other daily activities. The videos are challenging in their variety of appearance, non-stationary motion (e.g. accelerations or transient phenomena) and non-static appearance induced by camera motion or a changing motion appearance throughout the video. In this chapter we focus on dealing with such challenges as they often appear in the real-world.

non-stationary motion and automatically selects the most discriminative signal based on a self-estimated quality assessment. Extending beyond the existing repetition estimation dataset of [Levy and Wolf, 2015](#), we propose the new *QUVA Repetition* dataset for repetition estimation in video. The proposed dataset is more realistic and challenging as we lift the static and stationary assumptions. The experiments in this chapter focus on the task of repetition counting. On this task, we demonstrate that our method without any learnable component can outperform the deep learning-based method of [Levy and Wolf, 2015](#).

3.2 Related Work

Existing literature on repetition estimation in video commonly represent video as one-dimensional signals that preserve the repetitive structure of the motion. Once such a signal has been acquired, frequency information can be extracted by Fourier analysis ([Azy and Ahuja, 2008](#); [Cutler and Davis, 2000](#); [Pogalin et al., 2008](#); [Tsai et al., 1994](#)), peak detection ([Thangali and Sclaroff, 2005](#)), singular value decomposition ([Chetverikov and Fazekas, 2006](#)) or computational topology ([Tralie and Perea, 2018](#)). These methods can achieve compelling repetition counting results when considering video with static and stationary periodic motion.

The seminal work of [Cutler and Davis, 2000](#) uses normalized autocorrelation to obtain similarity matrices which can be used for estimating repetitions through Fourier analysis. [Pogalin et al., 2008](#) estimate the frequency of motion in a video by tracking an object, performing principal component analysis over the tracked regions, and again employing the Fourier-based periodogram. From the video’s spectral decomposi-

tion, the dominant frequencies can be identified by a peak detection algorithm followed by the non-trivial separation of fundamental and harmonic frequencies. While Fourier-based methods provide a good estimate of strongly periodic motion, they are unsuitable nor intended to deal with more realistic non-stationary repetition (see [Figure 2.6](#)).

Although strongly periodic motion has received serious attention, less effort has been devoted to non-stationary repetition in video. [Briassouli and Ahuja, 2007](#) use the Short-Time Fourier Transform for estimating the time-varying spectral components in video to distinguish multiple periodically moving objects. The filtering-based approach of [Burghouts and Geusebroek, 2006](#) draws inspiration from the time-causal filter bank proposed by [Koenderink, 1988](#) to detect quasi-periodic motion in video. Their method works online and shows remarkable results when filter response frequencies are tuned correctly. Inspired by their work, we adopt temporal filtering in the form of the continuous wavelet transform which uses multiple temporal scales to estimate repetition.

The deep learning method of [Levy and Wolf, 2015](#) distinguishes itself from all other work on repetition in video as it adopts a learning-based approach using a convolutional neural network. However, it is similar to our work in its counting-based evaluation over a large video dataset. Their general idea is to train a convolutional neural network for predicting the motion period in short video clips. As realistic training data is not available, the network is optimized on synthetic video sequences in which moving squares exhibit periodic motion of four motion types from [Pogalin et al., 2008](#). At test time, the method takes a sequence of video frames, performs explicit motion localization to obtain a region of interest, and finally predicts the period of motion by forwarding the localized frame crops through the convolutional network. The system is evaluated on the task of repetition counting and shows near-perfect performance on their own *YTSegments* dataset. These 100 videos are a suitable initial benchmark but as the majority of videos has a static viewpoint and exhibit stationary periodic motion, we propose a new dataset to set a more realistic benchmark. Specifically, our dataset better reflects reality by including more non-static and non-stationary examples as they would appear in practice.

Instead of considering repetition as their primary goal, various existing works leverage the presence of periodic motion for auxiliary tasks. [Belongie and Wills, 2006](#) exploit periodic human motion for 3D reconstruction of a scene. In a similar line of work, [Laptev et al., 2005](#) uses

sequence alignment on video containing periodic motion for stereo-camera correspondence. From a practical point of view, the presence of periodic motion also serves as a cue for action classification (Lu and Ferrier, 2004; Goldenberg et al., 2005) and can be beneficial for camera calibration (Huang et al., 2016).

3.3 Method

Grounded in the theoretical foundation of visual periodicity (Chapter 2), we here propose a method for estimating repetition in video with the application of counting the number of visually recurrent cycles. Our method for repetition estimation follows a three-stage approach as summarized in Figure 3.2. First, we localize the target instance in the scene, then we represent the target by a set of time-varying signals, and finally we perform time-frequency decomposition to estimate repetition and select the most discriminative flow representation using a novel selection algorithm. We will proceed by discussing our method’s details.

3.3.1 Flow-based Signals from Video

To deal with camera motion and to handle the wide variety in repetitions (see the 18 cases in Figure 2.4), we construct a diverse set of time-varying flow-based signals that we compute over the *motion foreground segmentation*. In particular, we measure the average-pooled flow field $\mathbf{F} = (F_x, F_y)$ and the differentials of the flow (2.11). We estimate $\nabla \mathbf{F}$ by measuring $\nabla_x F_x$ and $\nabla_y F_y$. All the differentials of the flow field are computed using Gaussian derivative filters with a large filter size to obtain a global measurement over the foreground segmentation. The final measurement is the average-pooled value over a small radius around the object’s center. The differential operators of the flow field yield four different measurements (the curl has only one direction; perpendicular to the screen), whereas there are two zeroth-order flow signals. In total, our approach considers six different signals.

For the cases of oscillating and intermittent motion observed from the side, $\nabla \mathbf{F}$ will deliver the strongest repetitive signal. The flow field \mathbf{F} will convey a stronger repetitive signal for the cases of constant motion appearance. In practice, it may be hard to select the most discriminative

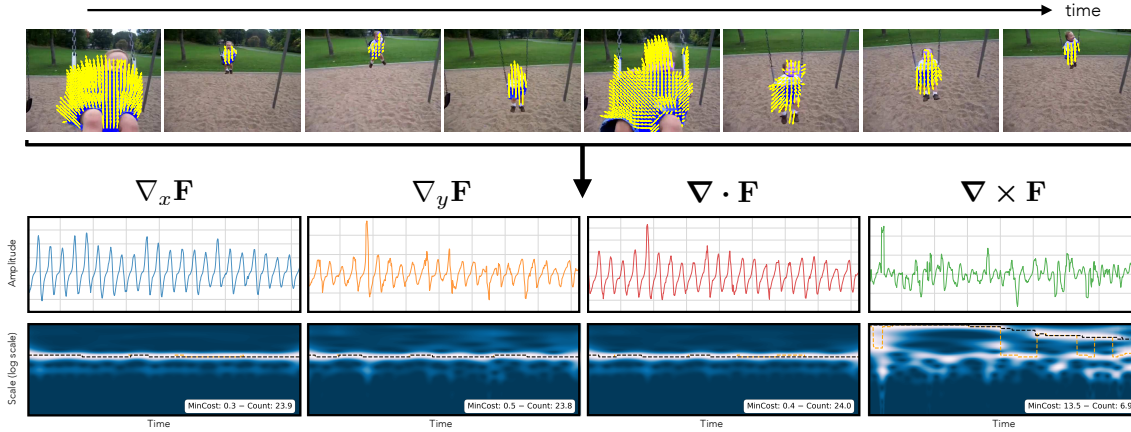


Figure 3.2. Overview of our method by illustration of an example. First, we segment the foreground motion (top row, blue masks) followed by optical flow computation (yellow arrows); then we extract zeroth- and first-order flow signals (4 out of 6 shown); and finally decompose them into a time-frequency spectrum using the continuous wavelet transform (bottom). In the bottom row, the dashed black lines denote the min-cost path whereas the orange lines indicate the maximum power path for counting by integration. Note that for this *oscillatory translation frontal view* case, $\nabla_x F_x$, $\nabla_y F_y$ and $\nabla \cdot \mathbf{F}$ give a good signal, as expected, whereas $\nabla \times \mathbf{F}$ gives a poor and dispersed signal associated with a high cost.

flow representation that best characterizes the video’s motion. We will return to this problem in [Section 3.3.4](#).

3.3.2 Continuous Wavelet Transform

At this point, we have represented the video’s salient foreground motion as a collection of six one-dimensional flow-based signals that we can further process using signal processing techniques.

Given a discrete signal h_n for timesteps $n = 1, \dots, N - 1$ sampled at equally spaced intervals δt . Let $\psi_0(\eta)$ be some admissible wavelet function, depending on the non-dimensional time parameter η . The continuous wavelet transform ([Grossmann and Morlet, 1984](#); [Mallat, 1989](#)) is defined as the convolution of h_n with a “daughter” wavelet generated by scaling and translating the wavelet function $\psi_0(\eta)$:

$$W_n(s) = \sum_{n'=0}^{N-1} h_{n'} \psi^* \left[\frac{(n' - n)\delta t}{s} \right], \quad (3.1)$$

where the asterisk represents the complex conjugate. By varying time parameter n and the scale parameter s , the wavelet transform generates a time-scale representation describing how the amplitude of the signal changes with time and scale. We use the Morlet wavelet, a complex exponential carrier modulated by a Gaussian envelope:

$$\psi_0(\eta) = \pi^{-1/4} e^{i\omega_0\eta} e^{-\eta^2/2}. \quad (3.2)$$

In all our experiments we set $\omega_0 = 6$ as it provides a good balance between time and frequency localization. Since the Morlet wavelet is complex, the wavelet transform $W_n(s)$ will also be complex. Therefore, it is useful to define the wavelet power spectrum or *scalogram* as $|W_n(s)|^2$ representing the time-frequency localized energy. As an example, [Figure 3.3](#) plots a non-stationary artificial signal and visualizes the corresponding wavelet power spectrum. We can observe that the scalogram is effective in revealing the signal’s non-stationary repetitive dynamics.

The resolution of the scalogram $|W_n(s)|^2$ is defined by the distribution of its scale parameter s . Following practical recommendations ([Torrence and Compo, 1998](#)), we will use a discrete scale set that is logarithmically distributed:

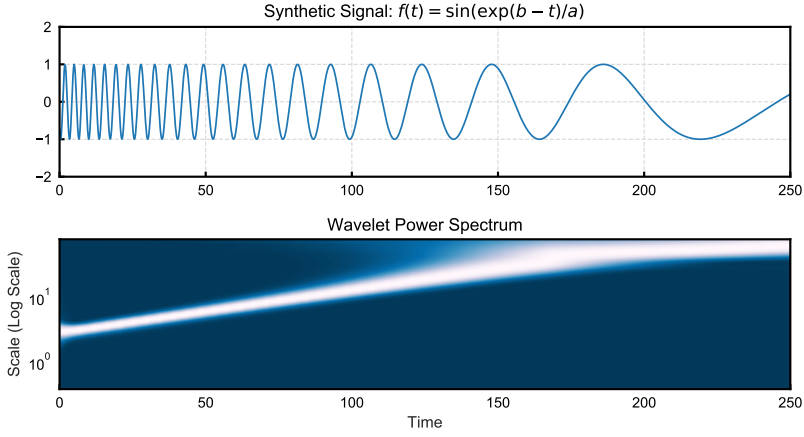


Figure 3.3. Exponential chirp signal and the corresponding scalogram obtained from the continuous wavelet transform. Note increasing scale (period) in the scalogram as the signal's frequency decreases.

$$s_j = s_0 2^{j\delta j}, \quad j = 0, 1, \dots, J \quad (3.3)$$

$$J = \delta j^{-1} \log_2 (N\delta t/s_0). \quad (3.4)$$

The smallest measurable scale s_0 and the number of scales J determines the range of the detectable frequencies. The smallest scale should be chosen such that the Fourier period of the wavelet is approximately $2\delta t$. For a moment in time, the scalogram's maximum power will give the wavelet scale s producing the strongest filter response. Often the temporal frequency associated with the scale s will be a more convenient measurement. Therefore, the wavelet scale can be converted to a temporal frequency. For a Morlet wavelet, the relationship between scale and wavelength is given by [Torrence and Compo, 1998](#):

$$\lambda = \frac{4\pi}{\omega_0 + \sqrt{2 + \omega^2}}, \quad (3.5)$$

where ω_0 corresponds to the non-dimensional frequency. For $\omega_0 = 6$ corresponds to $\lambda = 1.03s$ for the Morlet wavelet, thus having the attractive property of wavelet scale being almost identical to the wavelength. We use (3.5) to obtain the instantaneous frequency estimate for all discrete timesteps in the video.

To estimate non-stationary repetitions in a given video, we decompose each of the six flow-based signals into a time-frequency spectrum using the continuous wavelet transform. This yields six 2D time-frequency representations from which we will further estimate the repetitive contents in the video.

3.3.3 Repetition Counting

We will assess our method’s ability to estimate repetition on the task of *counting* action repetitions from video. Counting is a suitable task as the repetition count remains meaningful in the presence of non-stationarity whereas predicting the period of motion will not be meaningful. We assume there is only one dominant repetitive motion observable in the wavelet spectrum. This is reasonable as the foreground motion segmentation encourages temporal consistency. As discussed in the previous section, selecting the modulus maximum from the wavelet spectrum $|W_n(s)|^2$ for every timestep n gives a local frequency measurement. For a Morlet wavelet, the frequency estimate is approximately s_n^{-1} so we can obtain the *repetition count* by integrating over all discrete timesteps:

$$\hat{c} = \sum_n \delta t / s_n. \quad (3.6)$$

In practice, we will use the exact frequency estimate computed by (3.5). For a stationary periodic signal, the modulus maximum forms a horizontal ridge through time. We emphasize that the continuous wavelet transform enables the counting of non-stationary action repetitions using our approach. Consequently, and unlike most existing approaches, our method can handle accelerations or transient phenomena.

3.3.4 Min-Cost Signal Selection

Up until now, we have considered all six flow-based signals. However, we do not know which of the flow representations will best reveal the video’s recurrent motion. Therefore, wWhat remains is selecting the most discriminative signal out of the six. Our solution is a self-selection mechanism that prioritizes signals with local regularity in the time-frequency space. Specifically, we adopt a min-cost algorithm for finding the optimal path through the time-frequency space. We turn the wavelet power into a cost surface for optimization by simply inverting it: $1/|W_n(s)|^2$. Con-

sequently, traveling over a high-power region is associated with a low cost. As our goal is to characterize a signal by a single scalar-valued cost measure, we run a greedy min-cost pathfinding algorithm to determine the minimum cost required to traverse the spectrum from the beginning to the end. In other words, the algorithm effectively assigns a lower cost to paths with high local regularity, therefore favoring more strongly periodic signals. This is an attractive property as realistic video signals can be non-stationary but locally smooth. To make a final prediction we select the signal with minimum cost and its corresponding repetition count using (3.6). This concludes the discussion of our method for repetition estimation.

3.4 Datasets and Evaluation Metrics

For the experiments in this chapter, we consider two video datasets: the existing *YTSegments* and our new *QUVA Repetition* dataset, both collected for the purpose of evaluating repetition estimation in video.

3.4.1 YTSegments Dataset

To evaluate repetition counting in video, [Levy and Wolf, 2015](#) introduced a new video benchmark. The 100 videos downloaded from YouTube are for evaluation only as training the network is performed with synthesized videos. A wide range of actions appear in the videos: several sports, cooking and animal movement. Each video is temporally segmented such that the final clip contains a single repetitive action. The clips are annotated with the total repetition count. While the dataset serves as a good initial benchmark for repetition estimation, it is limited in terms of cycle length variation (*i.e.* non-stationarity), motion appearances and camera motion. As our goal is to evaluate our method under more realistic circumstances, we introduce a new video dataset that is more challenging in terms of non-stationarity, motion appearance, camera motion and background clutter.

3.4.2 QUVA Repetition Dataset

Our *QUVA Repetition* consists of 100 videos displaying a wide variety of repetitive video dynamics — including various kinds of sport, music-making, cooking, grooming, construction and animal behavior. The

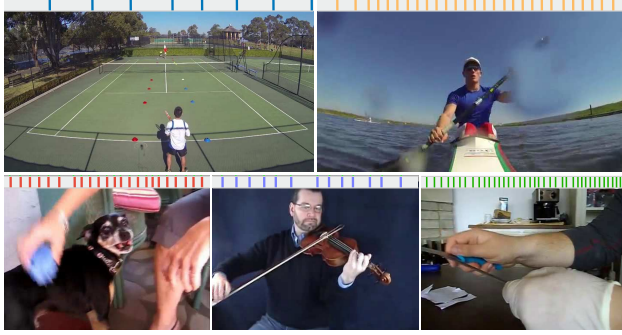


Figure 3.4. Annotation examples from the *QUVA Repetition* dataset. The time-lines with markers illustrate the individual cycle bound annotations that directly translate into the final repetition count. Note the diversity in motion appearance and cycle length variability within a video.

videos are collected from YouTube with an emphasis on creating a diverse collection of videos suitable for evaluating our method’s ability to deal with non-stationary motion, camera motion and significant evolution of motion appearance throughout a video.

After collecting the candidate videos, we adopt a multi-stage annotation process to obtain the final dataset. First, we asked two human annotators to label the temporal bounds of each interval containing at least four unambiguous repetitions. During this process, we found high inter-agreement between the annotators and keep the 100 intervals with the highest overlap to increase precision. Final video clips are obtained by temporal clipping of the intersection of the two intervals. As a result, some motion cycles may be partial, either at the beginning or end of the video. In the last round of annotation, we ask the annotators to mark all individual cycle bounds in the video clips (Figure 3.4) which also produces the final repetition count. We repeat this process for the *YTSegments* dataset to compare the inter-cycle length variability representing the level of non-stationarity.

The characteristics for both datasets are reported in Table 3.1 and Figure 3.5. Our videos have more variability in cycle length, motion appearance, camera motion and background clutter. Therefore, the increased difficulty in both appearance and temporal dynamics results in a more realistic benchmark for repetition estimation in the wild. In Figure 3.6 we present a collage of examples from both datasets.

Table 3.1. Dataset statistics of *YTSegments* and *QUVA Repetition*. The cycle length variation is defined as the average value of the absolute difference between the minimum and maximum cycle length divided by the average cycle length. To determine this, we annotate all individual cycle bounds for both datasets. The last two rows are also obtained by manual annotation.

	<i>YTSegments</i>	<i>QUVA Repetition</i>
Number of Videos	100	100
Duration Min/Max (s)	2.1/68.9	2.5/64.2
Duration Avg. (s)	14.9 \pm 9.8	17.6 \pm 13.3
Count Avg. \pm Std.	10.8 \pm 6.5	12.5 \pm 10.4
Count Min/Max	4/51	4/63
Cycle Length Variation	0.22	0.36
Camera Motion	21	53
Superposed Translation	7	27

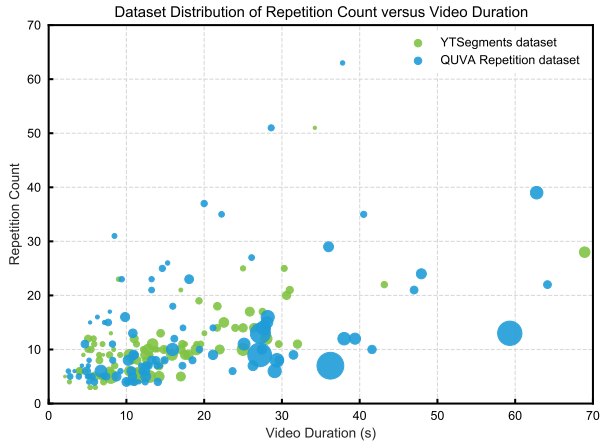
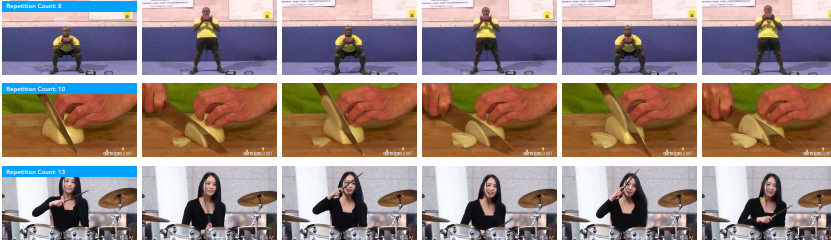


Figure 3.5. Distribution of repetition count versus video duration for the *YTSegments* and *QUVA Repetition* dataset. The radius of each datapoint is proportional to the cycle length variation of the video. Note the increased variability in non-stationarity and repetition count of our dataset.

YTSegments Dataset



QUVA Repetition Dataset

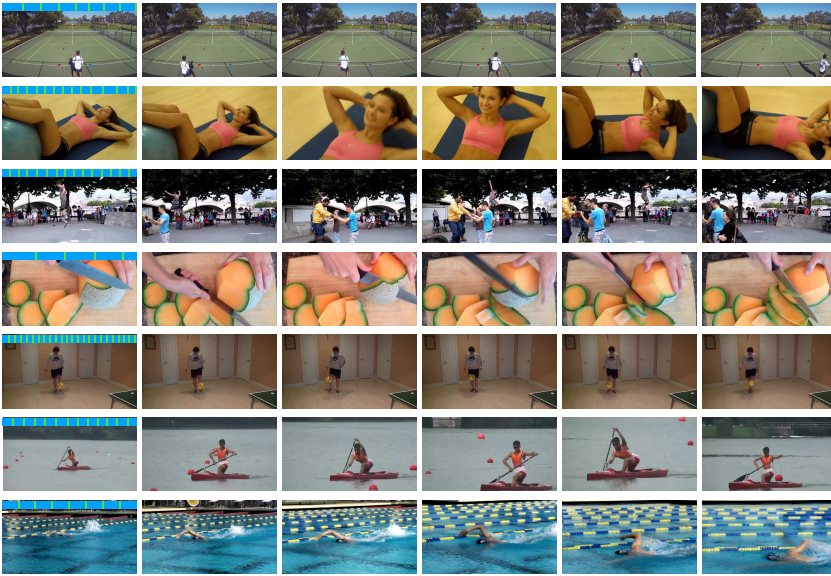


Figure 3.6. Examples from the *YTSegments* and *QUVA Repetition* datasets. The *YTSegments* dataset only contains a final repetition count annotation (indicated). Our dataset is additionally annotated with individual cycle bounds, suitable for determining the level of non-stationarity. The blue timeline (first frame) displays the individual cycle annotations for the given video. The final count is given by the number of full cycles. Note the increased cycle length variation and the higher difficulty of our dataset due to camera motion, occlusions and background clutter.

3.4.3 Evaluation Metrics

Given a set of N videos, we evaluate the performance between ground truth count c_i and the count prediction \hat{c}_i for all videos $i \in \{1, \dots, N\}$ of the dataset. For a fair comparison with prior work (Levy and Wolf, 2015) we report the mean absolute error (MAE). Furthermore, we also report the off-by-one-accuracy (OBOA) over the entire dataset:

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^N |\hat{c}_i - c_i| / c_i \quad (3.7)$$

$$\text{OBOA} = \frac{1}{N} \sum_{i=1}^N [|\hat{c}_i - c_i| \leq 1]. \quad (3.8)$$

We note that the mean absolute error (MAE) is preferred over the common mean squared error (MSE) as it is relative to the true count. Furthermore, to account for rounding errors and possible cycle cut-offs at both ends of the video, the off-by-one-accuracy is more suitable than the traditional accuracy. Both metrics will be used to assess the quality of our count predictions over both video datasets.

3.5 Experiments

In this section, we present experimental results on both video counting datasets. Before discussing the results, we will describe our method's implementation details and introduce the baselines for comparison.

3.5.1 Implementation Details

As discussed in Section 3.3.1, our method first segments out the foreground motion using the existing approach of Papazoglou and Ferrari, 2013. To account for incorrect segmentation masks we reuse the segmentation of the previous frame if the fraction of foreground pixels is less than 1% of the entire frame. We note that using a decoupled method for motion segmentation may not be optimal for handling repetitive motion, to which we will return in Chapter 4.

To estimate the flow field over the foreground segmentation masks, we rely on EpicFlow (Revaud et al., 2015). We also tested our method with the alternative flow based algorithms TV-L¹ (Zach et al., 2007) and

FlowNet 2.0.0 (Ilg et al., 2017) but found slightly better results with EpicFlow. Importantly, we found our method to be robust to the choice of optical flow algorithm.

Next, we compute the divergence and curl by first-order Gaussian derivative filters with a 13×13 filter size. We use a Morlet wavelet with logarithmic scales ($\delta j = 0.125$, $s_0 = 2\delta t$) based on the practical recommendations of Torrence and Compo, 1998. We limit the range of J corresponding to a minimum of four repetitions in the video. Before applying the wavelet transform, we apply temporal mean-filtering and linearly detrend the input signals. The temporal mean filter uses a window size of 7 time steps throughout all experiments. We keep all parameters fixed over all experiments and did not find it necessary to carefully tune them. Nonetheless, careful tuning or learning the optimal parameters may produce better results.

3.5.2 Counting Baselines

We compare our wavelet-based counting method to two existing methods suitable for counting repetition in video. We choose the method of Pogalin et al., 2008 to represent the class of Fourier-based approaches for repetition estimation. Based on the details outlined in the original paper, we reimplemented the method with our best efforts. Our reimplementation uses a more recent object tracker (Henriques et al., 2012) but is identical otherwise. The tracker is initialized by manually drawing a box on the first frame. Once the frequency has been determined, the repetition count is determined using the video duration a frame rate. Secondly, we compare our method with the more recent deep learning method of Levy and Wolf, 2015 using their publicly available code and pretrained model using the synthetic video sequences. We adopt all the hyper-parameters as suggested by the authors.

3.5.3 Temporal Filtering: Fourier versus Wavelets

Setup. The goal of our first experiment is to demonstrate the effectiveness of the continuous wavelet transform for counting repetitions in non-stationary signals. We compare the stationary Fourier-based periodogram with the time-scale representation given by the wavelet scalogram. To isolate the effect of frequency measurements, we generate *idealized* signals of the videos in our *QUVA Repetition* dataset. Specif-

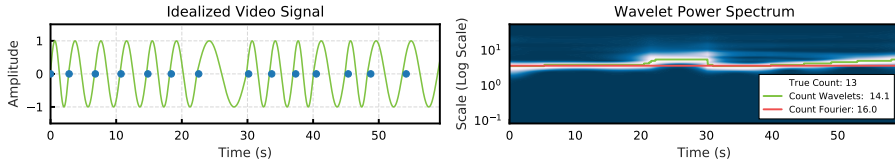


Figure 3.7. Idealized signal for a difficult non-stationary video displaying a violin player (see Figure 3.4). The blue markers indicate the cycle bounds, manually annotated for each video in our *QUVA Repetition* dataset. Note how the wavelet scalogram correctly exposes the rhythmic slowdown starting around the 20-second mark. On the right, the green line corresponds to the local frequency predictions from the scalogram whereas the red (straight) line indicates the stationary Fourier-based frequency measurement. This demonstrates the effectiveness of wavelet analysis for optical non-stationary video signals.

ically, we fit sinusoidal signals through the individual cycle bounds for each video to obtain simple 1D waveforms representing the video. Figure 3.7 shows an idealized signal example and the corresponding wavelet spectrum with count predictions. To compare with the Fourier-based measurement, we compute the periodogram, detect the maximum frequency peak and convert the corresponding frequency to a count using the video duration. This yields a repetition count prediction for both the stationary and non-stationary measurements that we evaluate over the entire dataset.

Results. From the results in Figure 3.8 we observe that wavelet-based counting significantly outperforms the periodogram on idealized signals. As expected, we observe that the Fourier-based measurements inevitably fail on videos with significant cycle length variation as they give a global frequency prediction. Wavelets naturally handle non-stationary repetition and are less sensitive to cycle length variability. We also tried adding a significant amount of Gaussian noise ($\sigma = 0.5$) to the signals and found this to have a slight negative effect on both methods but our method is more robust and overall conclusions remain the same. We observe that increased cycle length variation negatively affects Fourier-based counting. This is expected as it globally measures frequency and is unable to deal with non-stationarity. As wavelets naturally handle non-stationary repetition they are less sensitive to cycle length variability. This controlled experiment demonstrates the effectiveness of wavelets for repetition estimation under the assumption that a clear flow-based video signal is available.

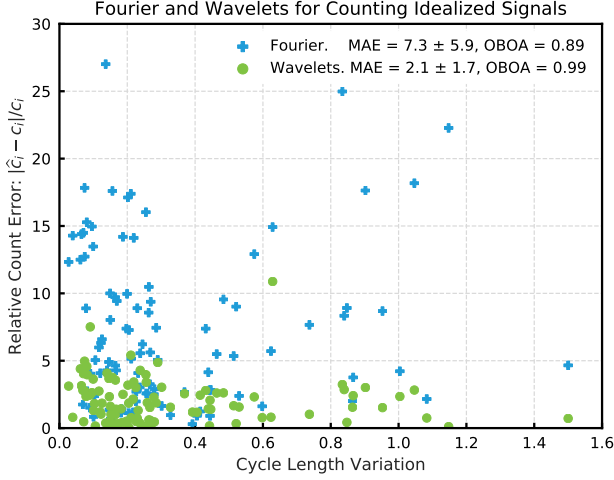


Figure 3.8. Fourier- versus wavelet-based repetition counting on idealized video signals from the *QUVA Repetition* dataset. Our wavelet-based method outperforms a Fourier-based baseline for 83 out of 100 videos. Increased cycle length variation results in notable error for Fourier measurements, whereas the time-localized wavelets are less sensitive to non-stationary repetition.

3.5.4 Value of Diverse Flow Signals

Setup. As wavelets prove to be effective for the counting task, we turn to the assessment of multiple flow-based signals for repetition counting. Motivated by the theory of [Chapter 2](#), the collection of six flow-based signals that we evaluate consists of: $F_x, F_y, \nabla_x F_x, \nabla_y F_y, \nabla \cdot \mathbf{F}, \nabla \times \mathbf{F}$. These are measured over the foreground segmentation and we will here evaluated each representation individually on the task of repetition counting. Like before, we report the MAE counting error on our *QUVA Repetition* dataset. To obtain a lower-bound on the error, we also select the best flow representation per video in an oracle fashion.

Results. The results in [Table 3.2](#) reveal that for the wide variability of repetitive appearance there is no one size fits all solution. The individual signals are unable to handle all variety of repetitive appearances by themselves, but their joint diversity provides a compelling lower-bound. The vertical flow \mathbf{F}_y is best overall and selected more often than the others by the oracle. We explain this bias towards vertical flow by the observation that our dataset contains many sports videos in which the

Table 3.2. Value of diversity in six flow-based signals on videos from our *QUVA Repetition* dataset. The last column denotes how often each signal is selected by the oracle. While the individual signals struggle to obtain good performance by themselves, exploiting their joint diversity is beneficial. We compare two different flow algorithms and use EpicFlow in following experiments.

(a) Results using EpicFlow as motion field estimate.			
	MAE	OBOA	# Selected
$\nabla \cdot \mathbf{F}$	44.9 ± 34.8	0.35	8
$\nabla \times \mathbf{F}$	44.9 ± 34.8	0.42	14
$\nabla_x F_x$	46.7 ± 30.8	0.24	12
$\nabla_y F_y$	42.7 ± 39.8	0.33	13
F_x	38.3 ± 31.4	0.40	19
F_y	32.9 ± 31.4	0.52	34
Oracle Best	10.5 ± 15.7	0.81	100

(b) Results using FlowNet 2.0 as motion field estimate.			
	MAE	OBOA	# Selected
$\nabla \cdot \mathbf{F}$	42.7 ± 36.0	0.33	9
$\nabla \times \mathbf{F}$	38.7 ± 32.7	0.38	17
$\nabla_x \mathbf{F}$	38.9 ± 28.4	0.43	7
$\nabla_y \mathbf{F}$	37.9 ± 34.0	0.38	15
F_x	37.9 ± 31.9	0.43	15
F_y	32.4 ± 44.0	0.54	37
Oracle Best	11.1 ± 16.2	0.80	100

gravity is often used as opposing force. Regardless of the bias, we observe that the all representations contribute to specific videos. Repeating this experiment on the *YTSegments* dataset with oracle signal selection achieves an MAE of 4.2 ± 5.2 . This underlines the gap in difficulty between both datasets.

3.5.5 Video Acceleration Sensitivity

Setup. In this experiment, we examine our method’s sensitivity to acceleration by artificially speeding-up videos. Starting from the *YTSegments* dataset, we induce significant non-stationarity by artificially accelerating the videos halfway. Specifically, we modify the videos such that after the midpoint frame, the speed is increased by dropping every second frame. What follows are 100 videos with a $2\times$ acceleration starting halfway in the video. We compare the resilience of our method with that of the deep learning approach (Levy and Wolf, 2015) which handles non-stationarity by predicting the period of motion in sliding-window fashion over the video. This experiment omits Fourier-based analysis, as by its nature, it will inevitably fail on this task.

Results. The bar chart of Figure 3.9 presents the counting results for both the original and artificially accelerated setting. On their own dataset with low cycle length variability, the neural network of Levy and Wolf, 2015 excels. However, artificial acceleration turns the results upside down, as our method suffers less in the presence of sudden acceleration halfway through the video. This reveals their sensitivity to acceleration, whereas our method deteriorates less. We note that due to acceleration the min-cost paths through the time-frequency space (Section 3.3.4) are more expensive than without acceleration. However, as the cost increases for all signals with approximately the same amount, the method remains capable to select the discriminative signal in most cases.

3.5.6 Comparison State-of-the-Art

Setup. We conclude the experiments of this chapter by carrying out a full repetition count comparison between our method and those of Pogalin et al., 2008 and Levy and Wolf, 2015 on both video datasets. Our method uses fixed hyper-parameters in all cases and relies on the min-cost signal selection algorithm for picking out the most discriminative flow representation.

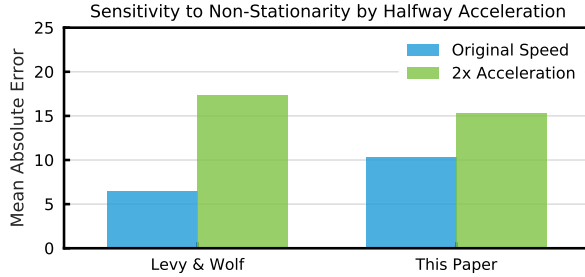


Figure 3.9. The effect of acceleration on the *YTSegments* dataset. The deep learning method of [Levy and Wolf, 2015](#) has difficulty dealing with non-stationary acceleration, whereas our method deteriorates less.

Results. The outcome of the final experiment is presented in [Table 3.3](#). For the *YTSegments* dataset, the method of [Levy and Wolf, 2015](#) performs best with an MAE of 6.5. Whereas our method scores 10.3, still significantly better than the Fourier-based approach of [Pogalin et al., 2008](#). Again, the results change when considering the more realistic and challenging *QUVA Repetition* dataset. The method of [Levy and Wolf, 2015](#) performs the worst, with an MAE of 48.2, which we attribute to the fact that their network only considers four motion types during training and therefore cannot handle the wider variety of appearances in our dataset. The Fourier-based method of [Pogalin et al., 2008](#) scores an MAE of 38.5, whereas we obtain an error of 23.2.

The system of [Levy and Wolf, 2015](#) is trained to handle motion periods ranging from 0.2 to 2.33 seconds. Our analysis shows that the error of their method increases for videos outside of this range. If we remove the 19 videos from *QUVA Repetition* outside of this range, the MAE of their method improves considerably to 28.3 ± 29.6 . As our method requires no learning, there is no limit on the cycle length range. On the same reduced set of videos, our method changes less and remains substantially better with an MAE of 17.6 ± 26.0 .

Roughly half of the videos in *QUVA Repetition* contain camera motion, whereas this fraction is much lower in the *YTSegments* dataset ([Table 3.1](#)). Evaluating the method of [Levy and Wolf, 2015](#) on the 53 videos with camera motion gives an MAE of 51.2 ± 54.5 , whereas it is 44.8 ± 68.4 for the videos without camera motion. Our method seems less sensitive to

Table 3.3. Comparison with the state-of-the-art on repetition counting for both datasets. The deep learning-based method (Levy and Wolf, 2015) achieves good results on their own dataset of relatively clean videos. On the more realistic and challenging *QUVA Repetition* dataset, our method improves considerably over existing work, be it based on Fourier or deep learning.

	<i>YTSegments</i>		<i>QUVA Repetition</i>	
	MAE ↓	OBOA ↑	MAE ↓	OBOA ↑
Pogalin et al., 2008	21.9 ± 30.1	0.68	38.5 ± 37.6	0.49
Levy and Wolf, 2015	6.5 ± 9.2	0.90	48.2 ± 61.5	0.45
Our method	10.3 ± 19.8	0.89	23.2 ± 34.4	0.62

camera motion as we obtain MAEs of 17.6 ± 23.8 and 21.3 ± 34.2 , without and with camera motion, respectively.

We also observe that all methods make a common mistake: over-counting videos with a factor of two. The similarity in these videos is that one full cycle contains the exact same motion first with one arm (or leg) followed by the other (*e.g.* walking lunges or swimming front-crawl). As the perceived motion is almost identical for both limbs, the estimated temporal dynamics are twice as fast. Again, the significant over-estimate of the motion frequency produces a large count error for all methods. Solving this problem is not easy, as repetition estimates in those cases are essentially also a correct prediction; however, the human annotators define salient motion as a full cycle with both limbs.

Overall we conclude that our method better handles the non-static and non-stationary video characteristics in our *QUVA Repetition* dataset while still performing reasonably well on the videos from *YTSegments*. In Figure 3.10, We highlight three visual examples including their segmentation mask and spectrogram. We observe that even when the foreground segmentation fails, our method is still able to predict the correct count.

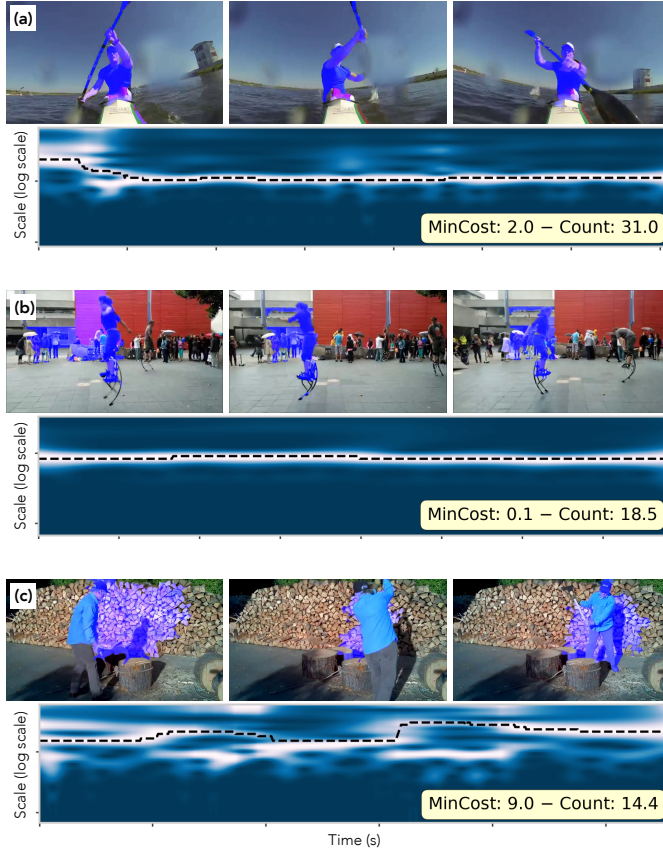


Figure 3.10. Results of our method for three examples. (a) The rower accelerates in the beginning of the video, which appears in the wavelet spectrum of signal F_x . Integrating over the max power path results in a repetition count of 31 whereas the true count is 30. Our method effectively handles the acceleration. (b) Stationary periodic motion superposed on translation. The video’s repetitive nature is evident from the F_y signal. We predict a repetition count of 18.5 whereas the true count is 18. (c) Change of viewpoint from *side* to *front* makes this video inevitably hard. Our method is unable to extract a good signal from the video. Note the partial continuity in the spectrum for $\nabla \times \mathbf{F}$ but distorted by the viewpoint changes. Our method predicts a repetition count of 14.4 whereas the true count is 16.

3.6 Conclusion

In this chapter, we have presented a method for estimating repetitive motion in video. Based on the theoretical foundation of [Chapter 2](#), our method considers a variety of flow-based representations that we estimate over the discriminative foreground motion. As repetitive motion is rarely perfectly periodic, we have considered the more suitable wavelet transform for estimating non-stationary recurrent motion. Our empirical evaluation on an existing dataset and our more realistic *QUVA Repetition* dataset indicates that our method can outperform a recent deep learning method on the task of repetition counting in video.

CHAPTER 4



IMPROVED REPETITION ESTIMATION

4.1 Introduction

In this chapter, we will present an improved method for repetition estimation in video. The current chapter largely maintains the original theory while making significant improvements to the previously introduced method for repetition estimation ([Chapter 3](#)). Specifically, we simplify our approach by removing the need for explicit motion segmentation prior to repetition estimation. Instead, we obtain a foreground motion segmentation directly from the wavelet filter responses densely computed over the motion maps. As the most discriminative motion representation is not known *a priori*, in [Chapter 3](#) we employed a self-quality assessment to select the representation best measurable. However, selecting a single most discriminative representation is inherently unsuitable for handling significant variations due to camera motion or motion evolution over the course of the video. Our solution is to combine the scalograms of all flow representations for robustness and to handle viewpoint changes.

As we have seen, estimating repetition in practice remains challenging. First and foremost, repetition appears in many forms due to its diversity motion types and motion continuity ([Figure 3.1](#)). Sources of variation in motion appearance include the action class, origin of motion and the observer's viewpoint. Moreover, the motion appearance is often *non-static* due to a moving camera or as the observed phenomena

This chapter is based on our IJCV 2019 publication ([Runia et al., 2019](#)).

develops over time. In practice, repetitions are rarely perfectly periodic but rather are *non-stationarity*. In [Chapter 3](#) we have seen that existing literature ([Levy and Wolf, 2015](#); [Pogalin et al., 2008](#)) generally assumes static and stationary repetitive motion. However, as reality is more complex, we here address some of the shortcomings of our previous method. Specifically, we will focus on the non-static and non-stationary aspect of visual repetition in real-world video.

To deal with the diverse and possibly non-static motion appearance in realistic video, the theory from [Chapter 2](#) implies representing the video with a mixture of first-order differential motion maps. We also concluded that for non-stationary temporal dynamics the fixed-period Fourier transform ([Cutler and Davis, 2000](#); [Pogalin et al., 2008](#)) is not suitable. As before, we handle complex temporal dynamics by decomposing the motion into a time-frequency distribution using the continuous wavelet transform. Improving upon the previous chapter, we increase our method’s robustness by improving the handling of camera motion. We achieve this by leveraging the information of all flow-based representations rather than selecting a single one using the min-cost path finding algorithm ([Section 3.3.4](#)). Finally, we remove the need for explicit tracking ([Pogalin et al., 2008](#)) or the previously used motion segmentation ([Papazoglou and Ferrari, 2013](#)). The improved method introduced here, segments the repetitive motion directly from the wavelet power spectra, therefore being better suitable for the task at hand, without reliance on external segmentation. We again evaluate on the task of repetition counting and demonstrate our method’s improved ability to handle dynamics viewpoint changes while outperforming existing work on the two video datasets.

4.2 Related Work

For a comprehensive overview of literature around repetition estimation we refer the reader to [Section 3.2](#). In this brief extension, we will focus on the localization part of repetitive motion in video as it is among the core improvements of this chapter.

Increased video complexity in terms of motion appearance, scene complexity and camera motion demands intricate spatiotemporal localization of salient motion. While many methods for periodic motion analysis incorporate some form of tracking or motion segmentation

(Polana and Nelson, 1997; Pogalin et al., 2008; Levy and Wolf, 2015), few approaches specifically address the challenge of repetitive motion segmentation. Goldenberg et al., 2005 estimate the repetitive foreground motion to use its center-of-mass trajectory for classifying human behavior. More closely related is the recent work of Lindeberg, 2017 in which scale selection over space and time leads to an effective temporal scale map. Inspired by this, we perform spatial segmentation of repetitive motion directly from the spectral power maps obtained through the continuous wavelet transform. This is appealing, as it connects localization to the temporal dynamics rather than relying on decoupled localization by motion segmentation (Tokmakov et al., 2017). Moreover, action localization itself is a task that remains challenging for real-world video.

4.3 Method

Building upon our previous efforts outlined in Chapter 3, we here present an improved method for estimating repetition in video. The method takes as input a sequence of RGB frames and outputs a temporal frequency distribution densely computed over both space and time. Subsequently, the spectral power distribution, which we obtain from the continuous wavelet transform, is used for repetition counting, motion segmentation or other frequency-based measurements. Our focus is on the general case in which moving objects may exhibit non-stationary periodicity or have a non-static appearance due to camera motion or repetition superposed on translation. Our method, summarized in Figure 4.1, comprises motion estimation and two consecutive filtering steps: first we spatially filter the motion fields to arrive at first-order differential geometric motion maps, and then we determine the video’s repetitive contents by applying the continuous wavelet transform densely over the motion maps. Task-dependent post-processing steps may give the desired output. As before, we focus on repetition counting since it enables straightforward evaluation of our method in the presence of non-stationary repetitions.

4.3.1 Differential Geometric Motion Maps

Given a sequence of video frames, we first estimate the motion between pairs of consecutive frames to obtain the motion field $\mathbf{F}(\mathbf{x}', t) = (F_x, F_y)$ for all timesteps. Next, the theory of visual periodicity (Chapter 2)

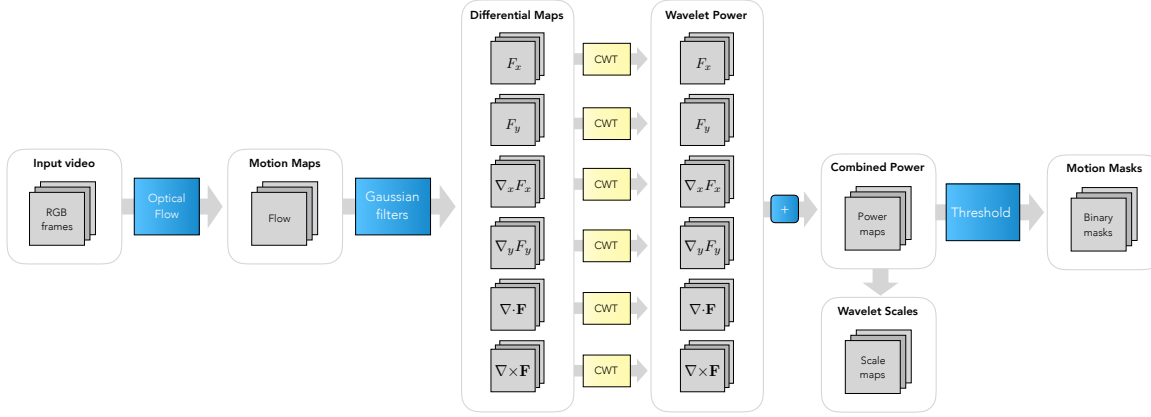


Figure 4.1. Overview of our method for repetition estimation in video. Given an input video as RGB frames we first estimate the motion between consecutive frames using optical flow. We perform spatial Gaussian filtering to obtain six (differential) motion representations. Next, we apply the continuous wavelet transform (CWT) through temporal convolution over all six representations individually. We combine all power maps by summation to arrive at a single power map for a moment in time. Finally, we spatially segment repetitive motion by mean-thresholding of the power maps. To estimate repetition, we median-pool the wavelet scales over the motion segmentation producing an instantaneous frequency measurement.

implies decomposition of the motion field into the primitive first-order differentials. For a moment in time t , we compute the differential motion maps by spatially convolving the flow field with first-order Gaussian derivative filters:

$$G^x(\mathbf{x}'; \sigma) = -\frac{x}{2\pi\sigma^4} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \quad (4.1)$$

$$G^y(\mathbf{x}'; \sigma) = -\frac{y}{2\pi\sigma^4} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right), \quad (4.2)$$

where σ denotes the spatial scale parameter and image coordinates are given by $\mathbf{x}' = (x, y)$. Through convolution with Gaussian kernels we obtain the first-order spatial derivatives $\nabla_x F_x, \nabla_y F_x, \nabla_x F_y$ and $\nabla_y F_y$ for a moment in time. Given the spatial partial derivatives of the motion, we compute $\nabla \cdot \mathbf{F}$ and $\nabla \times \mathbf{F}$ using the 2D equivalents of (2.5) and (2.8). For the 2D case, curl is a single-component vector field perpendicular to the image plane whereas the divergence is a scalar field. To effectively handle all cases of repetitive motion (Figure 2.4), we compute six motion maps for each video frame:

$$\{\nabla \cdot \mathbf{F}, \nabla \times \mathbf{F}, \nabla_x F_x, \nabla_y F_y, F_x, F_y\} \quad (4.3)$$

Periodicity in $\nabla \cdot \mathbf{F}$ or $\nabla \times \mathbf{F}$ will only occur for the frontal view. For oscillatory or intermittent motion from the side view, $\nabla_x F_x$ and $\nabla_y F_y$ will produce the strongest periodicity while the zeroth-order flow field F_x and F_y will deliver a stronger response for the cases of repetitive appearances at constant motion.

Figure 4.2 displays an example video frame with four out of six motion maps (two are omitted for space). The six motion maps represent the video for each moment in time and address the diversity in repetitive motion. Unlike in the previous chapter, we leverage the entire motion representations as we do not average pool the flow over the foreground segmentation as was described in Section 3.3.1. In our experiments, we will evaluate the individual and joint representative power associated with the motion maps. *A priori* we do not know which motion we are dealing with, to which we return later by combining the temporal responses of all motion maps.

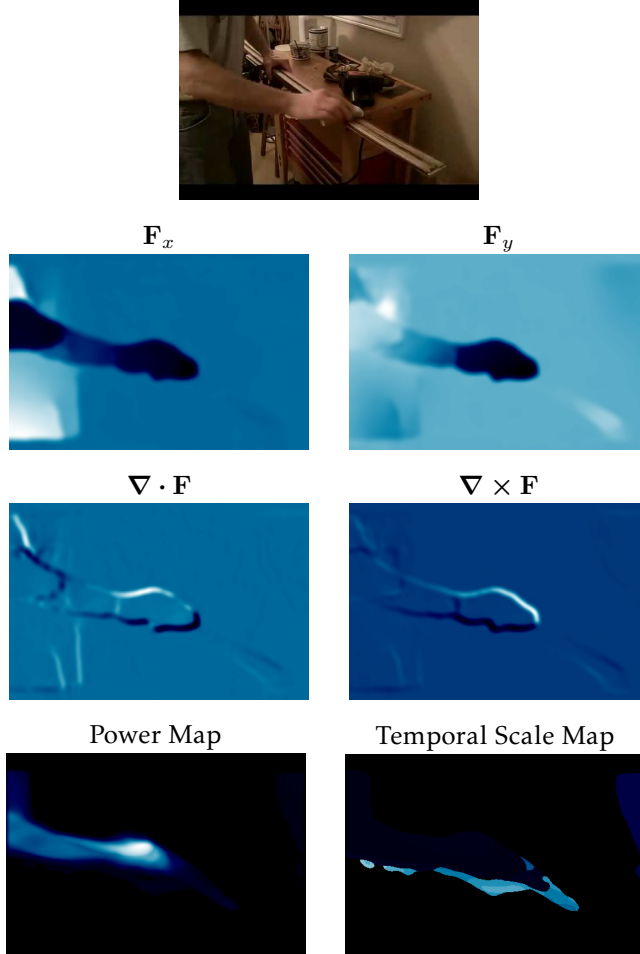


Figure 4.2. Intermediate motion maps for a video displaying a *man brushing wood* from the *QUVA Repetition* dataset. We perform wavelet filtering over six motion maps ($\nabla_x \mathbf{F}_x$ and $\nabla_y \mathbf{F}_y$ are omitted). Notice how the regions with repetitive motion appear in the wavelet power maps. By thresholding the wavelet power map with the mean power we obtain a repetitive motion map. The temporal scale maps directly relate to the motion frequency.

4.3.2 Dense Temporal Filtering

So far, we have only considered spatial filtering to obtain the motion maps for a moment in time. Here we consider time and proceed by temporal filtering of the motion maps to estimate the video's repetitive motion. This is where the current method deviates from the previous work. In [Chapter 3](#), we relied on the motion representations but performed average pooling over the foreground motion segmentation obtained separately from [Papazoglou and Ferrari, 2013](#). The average-pooled values over time construct a one-dimensional signal acting as a surrogate for the dynamics in a particular motion map. Spectral decomposition for each of the signals led to six, possibly contrasting, time-frequency estimates. To select the most discriminative representation, we proposed a self-quality assessment by finding a min-cost path through the spectrograms.

During the experiments of the previous chapter, we found two problems with this approach: (1) the decoupled motion segmentation may not be optimal for estimating repetitive motion dynamics, and (2) average pooling over the foreground motion mask discards most information and is unable to deal with multiple moving parts. Here, our improved method addresses both of these problems by dense temporal filtering over all locations in the motion map rather than operating on the average-pooled signals. Spatially dense estimation of the local spectral power enables us to localize regions likely containing repetitive motion.

The temporal filtering can be implemented in several ways, for example, as Fourier transform through temporal convolution. However, to handle non-stationary video dynamics in our dataset, we adopt the continuous wavelet transform by convolution to obtain a spectral decomposition. For convenience, we briefly summarize the background on continuous wavelets from [Section 3.3.2](#). Given a discrete signal h_n for timesteps $n = 1, \dots, N - 1$ sampled at equally spaced intervals δt . Let $\psi_0(\eta)$ be some admissible wavelet function, depending on the non-dimensional time parameter η . The continuous wavelet transform can be defined as convolution of h_n with a wavelet generated by scaling and translating the wavelet function $\psi_0(\eta)$:

$$W_n(s) = \sum_{n'=0}^{N-1} h_{n'} \psi^* \left[\frac{(n' - n)\delta t}{s} \right], \quad (4.4)$$

By variation of the time parameter n and the scale parameter s , the wavelet transform can generate a time-scale representation describing how the amplitude of the signal changes with time and scale. Similar to the previous chapter, we will use a Morlet wavelet (3.2) in all experiments. We consider the wavelet power spectrum $|W_n(s)|^2$ representing the time-frequency localized energy for estimating repetitions from video (see Figure 3.3 for an example signal and its power spectrum).

4.3.3 Combining Spectral Power Maps

Using the wavelet transform (4.4), we compute the time-localized frequency estimates by temporal convolution densely over the six individual motion representations. Unlike the previous chapter, where the wavelet transform was applied over an average-pooled 1D flow-based signal, we here maintain a spatial distribution of time-varying spectral power. In other words, for each of the six differential motion maps, we obtain time-varying maximum *power map* and *scale map*. The power map contains the spatial distribution of maximum wavelet power over all temporal scales; the scale map holds the temporal scales corresponding to the wavelets with maximum power. What remains is combining the wavelet responses from all motion representations to replace the min-cost algorithm.

Rather than selecting the single most discriminative representation (Section 3.3.4), we combine the spectral power maps by summation on a per-frame basis. To illustrate this, we visualize four (out of six) individual power maps and their combined response in Figure 4.3. Summation of the spectral power maps translates into a number of attractive properties. Most importantly, the motion maps with the strongest repetitive appearance will contribute most to the final power map whereas weakly periodic motion representations will have a negligible contribution. This effectively serves as a dynamic selection of the most discriminative motion representation. Moreover, as the spectral power is localized in time, the relative contribution per motion representation will be evolving over time. This is appealing because the perceived motion field can be non-static in video due to camera motion or change in motion type.

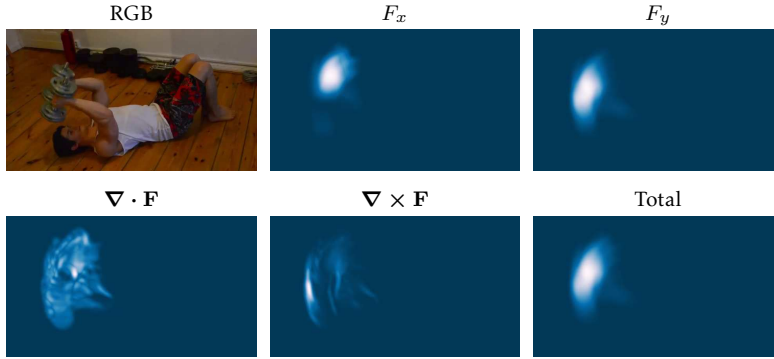


Figure 4.3. Video displaying a *man lifting weights* from our video dataset and its corresponding wavelet power maps for individual representations (we omit $\nabla_x F_x$ and $\nabla_y F_y$). In the bottom right, the total wavelet power obtained through the summation of all six responses. We normalize the power maps for displaying purpose. The vertical flow and curl produce the power maps with the largest norm for this moment in time. Summation of the individual power map effectively combines the filter responses.

4.3.4 Spatial Segmentation

The combined wavelet power map yields a time-varying spatial distribution of spectral power over all motion representations, whereas the corresponding effective scale map relates to the temporal scale with maximum spectral power. We propose to use the spatial distribution of spectral power for *localization* of the regions with strongest repetitive appearance by means of segmentation. Afterwards, we use the scale map to infer the dominant temporal scale.

The spatial segmentation of repetitive motion is performed in a straightforward manner. For a moment in time, we simply *mean-threshold* the combined wavelet power map to obtain a binary segmentation mask associated with regions containing significant spectral power. More precisely, the wavelet-based motion segmentation will attend to regions in which the maximum spectral power over all temporal scales is significant. The bottom row of [Figure 4.2](#) illustrates this by displaying the combined power map and corresponding scale map. Performing motion segmentation directly from the spatial distribution of spectral power is appealing as it couples the localization and subsequent frequency measurements. Our experiments will verify this claim as we compare our localization approach with specialized motion segmentation methods.

We would like to emphasize that our segmentation method leaves the door open for multiple repetitively moving objects whereas most state-of-the-art segmentation methods assume a single dominant foreground motion (Tokmakov et al., 2017).

4.3.5 Repetition Counting

Counting the number of repetitions in a video remains similar to the previous chapter. To obtain an instantaneous frequency estimate of the salient motion, we median-pool the temporal wavelet scales over the segmentation mask. Median-pooling is preferred over mean-pooling as it is relatively robust to outliers and is more likely to produce a better estimate of the dominant frequency. The corresponding temporal wavelet scale is subsequently converted to an instantaneous frequency using (3.5). For a moment in time, this will deliver a frequency estimate for the localized repetitive motion. Counting the number of repetitions follows temporal integration of the consecutive frequency measurements with the temporal sampling spacing inferred from the video’s frame rate.

We emphasize our method’s ability to count the number of cycles in non-stationary video. For a stationary periodic signal, the median-pooled temporal scales will be constant over time, while non-stationarity motion produces time-varying frequency estimates. Although the videos considered in our experiments are temporally segmented, the time-localized wavelet responses could also be used for temporal localization of repetitive actions. We further note that the median-pooling of the aggregated power maps could potentially be used for handling video with multiple recurrent moving parts.

4.4 Implementation Details

Before turning to our experiments, we will discuss the experimental details. As in the previous chapter, we will compare our method to the existing methods of Pogalin et al., 2008 and Levy and Wolf, 2015 on the datasets introduced in Section 3.4.

4.4.1 Optical Flow

Our method takes two consecutive video frames as input and first estimates the motion using optical flow. As the quality of motion estimation

may be important, we measure our method’s sensitivity to three flow estimation methods. To evaluate a more traditional flow estimation method we choose TV-L¹ (Zach et al., 2007). This variational based method is still competitive with more recent methods. Current state-of-the-art motion estimation methods all use convolutional neural networks for the purpose. We compare the deep learning based methods EpicFlow (Revaud et al., 2015) and FlowNet 2.0 (Ilg et al., 2017). Both deep networks are trained on large synthetic video datasets to estimate the motion in complex video. As default we use FlowNet 2.0 but the results with EpicFlow are similar.

4.4.2 Motion Segmentation

Complex videos with background clutter or camera motion demand segmentation of the foreground motion prior to further analysis. Our method directly performs localization from the densely computed wavelet power and we compare this to specialized motion segmentation methods. The fast video segmentation method of Papazoglou and Ferrari, 2013 is chosen as classical approach and was also used in Chapter 3. This approach separates foreground objects from the background in a video by combining motion boundary detection followed by segmentation refinement. We also evaluate the more recent deep learning-based method of Tokmakov et al., 2017. The method trains a two-stream convolutional neural network with a long-short term memory (LSTM) module to capture the evolution over time. The network parameters are optimized using the large FlyingThings 3D dataset (Mayer et al., 2016). To refine the motion masks from the trained networks, a conditional random field is applied for refinement. For both methods we use the official implementations made available by the authors. While both methods generally attain high-quality segmentations, we observed that segmentation fails completely for some more difficult frames (either all or none pixels selected as foreground). To remedy incorrect segmentation masks we reuse the previous segmentation if the fraction of foreground pixels is below 1% of the entire frame.

4.4.3 Differential Geometric Motion Maps

To compute the motion maps we perform spatial filtering by first-order Gaussian kernels. The filtering is implemented in PyTorch (Paszke et al.,

2019) and runs in large batches on the GPU to accelerate computation. Spatial convolution is performed with $\sigma = 4$ for all experiments. We also evaluated $\sigma = \{2, 8, 16\}$ but found only marginal variation in performance. In practice, a combination of multiple spatial scales may produce best results at increased computational cost. Once the spatial first-order derivatives $\nabla_x F_x, \nabla_y F_x, \nabla_x F_y$ and $\nabla_y F_y$ have been obtained through convolution, the differential motion maps are computed as specified in Section 4.3.1.

4.4.4 Continuous Wavelet Transform

We use the continuous wavelet filtering implementation as outlined in Torrence and Compo, 1998. Different from Chapter 3, we now perform temporal filtering on the GPU resulting to achieve a considerable speed-up. This enables us to apply the wavelet transform in large batches over all spatial locations in the video. As previously mentioned, we use a Morlet wavelet ($\omega_0 = 6$) with logarithmic scales ($\delta j = 0.125, s_0 = 2\delta t$). We limit the range of J corresponding to a minimum of four repetitions by setting s_{\min} and s_{\max} accordingly in (3.3) and (3.4). Depending on the video length, there are typically between 50 and 60 temporal scales levels. When compute budget is limited, computational efficiency can be improved by pruning the filter bank with scale selection, for example using the maximum response of a Laplacian filter (Lindeberg, 2017). Alternatively, learning the distribution of scales could be used to effectively prune the filter bank scales.

4.4.5 Repetition Counting

The instantaneous frequency estimates are obtained from the dense wavelet power by pooling over the motion foreground mask. As detailed in Section 4.3.5, the frequencies are integrated the discrete timesteps to arrive at a final repetition count. To remove frequency estimate outliers inconsistent with adjacent frames, we apply a median filter of 9 timesteps (frames) to increase local smoothness. This gives a minor improvement on both video datasets. The final repetition count predictions are not rounded to the nearest integer, hence evaluation metrics may deviate slightly due to incomplete cycles.

4.5 Experiments

We perform experiments to show the effectiveness of our method on the task of counting repetitions in video. Some experiments resemble those of [Chapter 3](#) but remain relevant as the method is different. Before evaluating our full method, we demonstrate our method’s ability to handle dynamic viewpoint changes and the wavelet’s effectiveness for dealing with acceleration recurrent motion.

4.5.1 Viewpoint Invariance

Setup. The theory of repetition considers two viewpoint extremes ([Figure 2.4](#)). In the first experiment, we evaluate our method’s ability to handle a continuous transition from one viewpoint extreme to the other. The designated mechanism for this is the use of multiple motion representations and the aggregation of their spectral power obtained from the continuous wavelet transform (see [Section 4.3.3](#)). To evaluate this, we consider a controlled experiment in which we synthesize a video clip from 3D modeled data in Blender. This enables full control over the object’s motion and the viewpoint. Specifically, we choose to build a simple 3D scene containing a ball periodically bouncing on the floor as displayed in the top row of [Figure 4.4](#). Initially, the camera captures the bouncing ball from the side view but after a number of full motion cycles, the camera smoothly transitions to frontal view (corresponding to case 3 to case 6 in [Figure 2.4](#)). We measure the median-pooled vertical flow and divergence over the foreground region to obtain two time-varying signals. The spectral power for both signals is individually estimated using the continuous wavelet transform, after which we combine the power by summation. In addition to a synthetic experiment, we also include the result of a real-world video with significant dynamic viewpoint change ([Figure 4.5](#); top row).

Results. [Figure 4.4](#) and [Figure 4.5](#) plot the two median-pooled flow signals and their joint wavelet power obtained by summation. Initially, as the moving object is captured from the side view, vertical flow is best measurable. Upon the viewpoint transition, vertical flow vanishes while the divergent flow becomes dominant. As a result of the camera motion, the measurement of the spectral power for both individual signals will only give a strong response for either the first or second

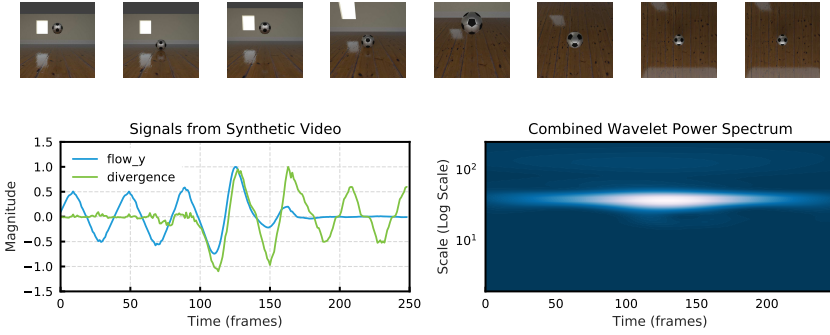


Figure 4.4. *Top:* synthesized video sequence for a controlled experiment on the influence of viewpoint relative to the motion. This video clip shows a 3D modeled scene containing a bouncing ball. At the midpoint of the animation, the camera smoothly transitions from side view to frontal view. *Bottom Left:* the time-varying magnitude of vertical flow and divergence measured over the foreground segmentation. Initially, the vertical flow is dominant and divergence is negligible. This reverses with the viewpoint transition. *Bottom Right:* the combined wavelet spectrum of both signals. Notice the spectrum’s invariance to viewpoint change as a result of wavelet power summation.

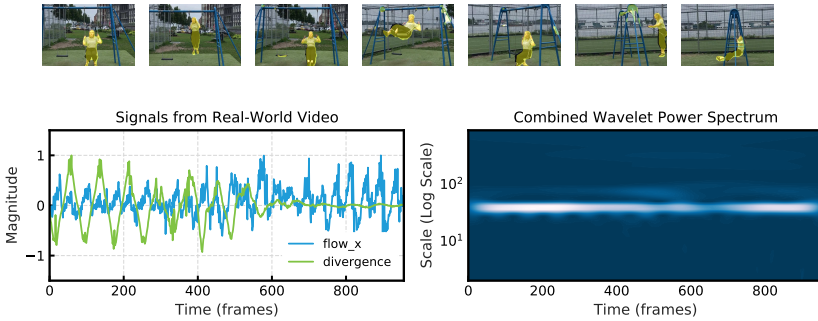


Figure 4.5. *Top:* example of dynamic viewpoint change for a real-world video. *Bottom Left:* the time-varying magnitude of horizontal flow and divergence measured over the foreground segmentation. *Bottom Right:* the combined wavelet spectrum of both signals. Again, by combining multiple representations through summation of the wavelet spectra, we obtain a representation that is invariant to viewpoint changes.

half of the video. However, the summation of the spectra gives a clear measurement over the complete video as is apparent from the combined wavelet power spectrum. This illustrates our method’s ability to handle viewpoint changes by the combination of the wavelet power contained in multiple motion representations. By summation of the spectra, the best measurable motion representation will naturally give the largest contribution to the combined power. Therefore, this mechanism acts as a replacement of the global representation selection used in [Chapter 3](#) by dynamically leveraging information in all representations.

4.5.2 Video Acceleration Sensitivity

Setup. We now repeat the experiment previously conducted in [Section 3.5.5](#) with our improved method. To recapitulate: we examine our method’s sensitivity to acceleration by artificially speeding-up videos. Starting from the *YTsegments* dataset, in which most videos exhibit strong periodic motion, we induce significant non-stationarity by artificially accelerating the videos halfway. More precisely, we modify the videos such that after the midpoint frame, the speed is increased by dropping every second frame. What follows are 100 videos with a $2\times$ acceleration starting halfway. We compare against the deep learning method of [Levy and Wolf, 2015](#) which handles non-stationarity by running the period-predicting convolutional neural network in sliding-window fashion over the video.

Results. The bar chart of [Figure 4.6](#) presents the mean absolute error in both original and accelerated setting. On their own dataset, the system of [Levy and Wolf, 2015](#) slightly outperforms our method. Acceleration reverses the results as our method suffers less and obtains a lower error on the accelerated videos. It reveals their sensitivity to acceleration, whereas our method deteriorates less. This shows the effectiveness of wavelets for dealing with non-stationarity in realistic videos. To illustrate how our method deals with midpoint acceleration, we also plot ([Figure 4.7](#)) the count increments and cumulative counts throughout the video. We noticed there is a distinct increase in count increments per timestep when upon enabling acceleration. This is observed for most videos in the dataset. We note that this could potentially be beneficial for detecting acceleration of motion or the temporal localization of transient phenomena in video.

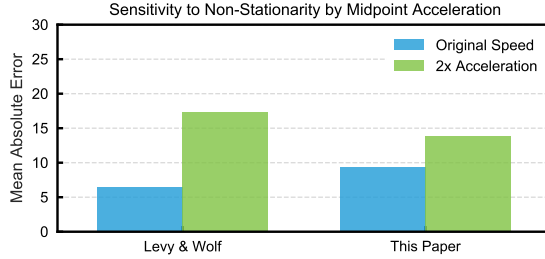


Figure 4.6. The sensitivity to midpoint acceleration on the *YTSegments* dataset. Our method increases 4.4 in mean absolute error whereas the method of [Levy and Wolf, 2015](#) rises with 10.8 points. The deep learning method has difficulty dealing with non-stationary acceleration, whereas our method is more robust due to use of the continuous wavelet transform.

4.5.3 Motion Segmentation

Setup. In this experiment we investigate the effectiveness of the motion segmentations obtained directly from the wavelet power for repetition estimation. We visually compare the motion segmentations and assess whether replacing our localization mechanism with a state-of-the-art motion segmentation method improves repetition estimation performance. We keep the method identical except for the segmentation method to obtain a motion mask. In addition to our wavelet-based motion segmentation to obtain the discriminative motion mask we assess our method’s performance (1) without any form of localization, *i.e.* full-frame; (2) the video segmentation method of [Papazoglou and Ferrari, 2013](#); and (3) the deep learning approach of [Tokmakov et al., 2017](#).

Results. We first visually compare the three different motion segmentation methods in [Figure 4.8](#). For most videos, our method is able to localize the repetitive motion. As the emphasis of our work is on repetition estimation, where the segmentation masks are a byproduct, the state-of-the-art specifically devoted to foreground motion segmentation naturally produce the most accurate results measured as lowest intersection-over-union error with respect to the ground truth mask. However, our intention is to obtain a motion mask best suitable for repetition estimation which not necessarily overlaps with the foreground motion. By mean-thresholding the wavelet power maps, our method seems to emphasize on regions with most discriminative repetitive mo-

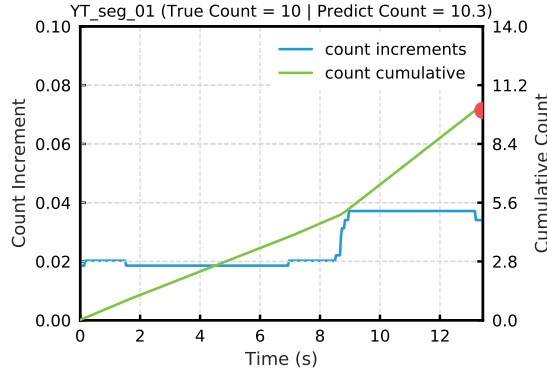
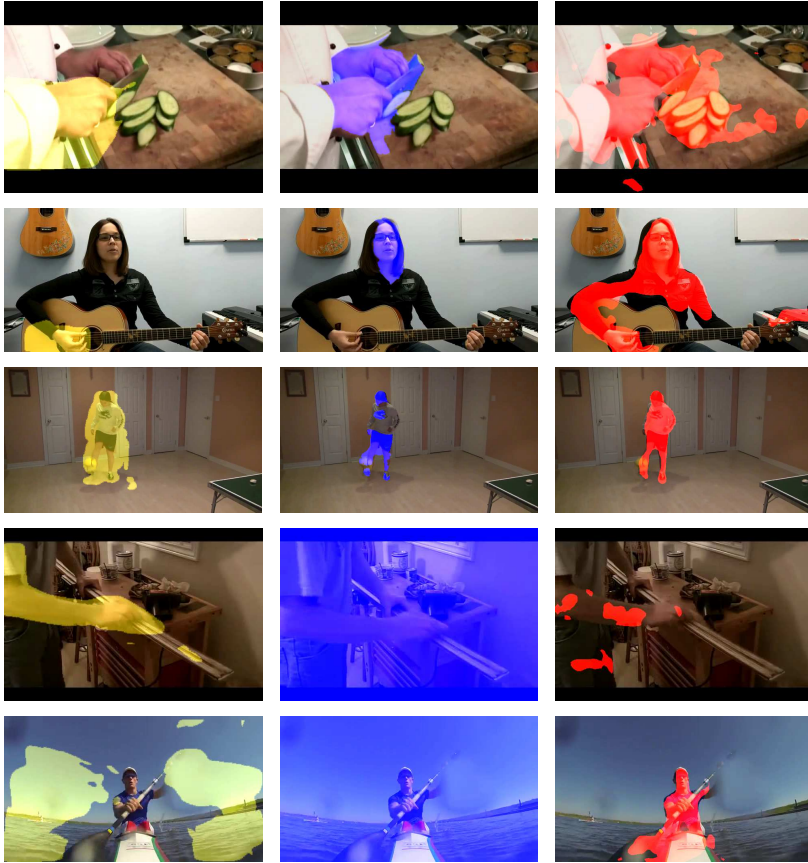


Figure 4.7. Count increments and cumulative count over time for the first video of *YTSegments* with midpoint acceleration. The red marker on the right corresponds to the ground truth count. Note how the increase in speed around 9 seconds is clearly reflected in the count increments.

tion. This is best recognizable from the two bottom rows where the motion segmentation includes background regions that periodically change due to the motion. If a high intersection-over-union overlap with respect to the ground truth foreground motion mask is desired, we observe a number of failure cases. For the *rower* (bottom row), the periodic response of the paddle movement yields a significantly stronger wavelet power than the body itself, hence the body is excluded from the segmentation mask due to mean-thresholding of the wavelet power. In case of *football keep-ups* (third row), the dominant repetitive motion is the football moving up-and-down but the actor also rotates around its axis which is not clear in the static images. However, the oscillating ball yields the strongest response and our localization approach wrongly includes the person’s torso. The threshold is currently fixed to the mean wavelet power, setting adaptively could improve the segmentation masks.

In [Table 4.1](#) we report quantitative results of our method with different motion segmentation methods. In terms of repetition counting, our localization mechanism produces significantly better results than the existing motion segmentation methods. We visualize the segmentation masks and corresponding counts for three examples at the end of this chapter ([Figure 4.9](#)). For our method, this convincingly demonstrates that the segmentation directly obtained from the wavelet spectrum are more suitable than decoupled motion segmentation approaches that are state-of-the-art on their own domain.



(a) Our work

(b) Papazoglou *et al.* 2013

(c) Tokmakov *et al.* 2017

Figure 4.8. Comparison of different motion segmentation masks. In most cases, our method succeeds to spatially segment the repetitive motion. In comparison to methods specifically devoted to the task of motion segmentation, our masks are less precise. However, as our numerical evaluation shows, our segmentation masks are more suitable for the task of repetition estimation. The most informative repetitive cues do not necessarily overlap with the foreground motion. In the last example, the regions through which the paddles moves produce the strongest repetitive response.

Table 4.1. Repetition counting results of our method with different motion segmentation mechanism. While the state-of-the-art motion segmentation methods produce visually excellent results, their segmentations are suboptimal for the task of repetition estimation. This is expected as the most discriminative repetitive cues are not always contained in the foreground motion. See [Figure 4.8](#) for a visual comparison of segmentation masks.

Localization method	<i>YTSegments</i>		<i>QUVA Repetition</i>	
	MAE ↓	OBOA ↑	MAE ↓	OBOA ↑
Full-frame	46.0 ±67.2	0.28	60.8 ±49.4	0.22
Papazoglou and Ferrari, 2013	13.1 ±20.3	0.78	42.6 ±49.2	0.44
Tokmakov et al., 2017	21.6 ±57.2	0.76	38.9 ±39.2	0.42
Our method	9.4 ±17.4	0.89	26.1 ±39.6	0.62

4.5.4 Comparison to the State-of-the-Art

Setup. In this experiment, we perform a full comparison on the task of repetition counting for both video datasets. We compare against the Fourier-based method of [Pogalin et al., 2008](#) and the deep learning approach of [Levy and Wolf, 2015](#).

Results. The full count evaluation is presented in [Table 4.2](#). On their own *YTSegments* dataset, the method of [Levy and Wolf, 2015](#) performs best with an MAE of 6.5, where our method achieves a comparable error of 9.4 and near-identical off-by-one accuracy. Despite the stationary nature of most videos in this dataset, the Fourier-based approach of [Pogalin et al., 2008](#) performs unfavorably compared to all other methods. A closer look at the intermediate steps of the Fourier-based method reveals the inferior performance is largely due to tracking failures and the Fourier transform’s sensitivity to such failures. The neural network is better able to handle imprecise localization results.

The results change dramatically when considering our challenging *QUVA Repetition* dataset; notably the deep learning approach of [Levy and Wolf, 2015](#) now performs the worst, with an MAE of 48.2. This could possibly be explained by the fact that their network only considers four motion types during training or the convolutional network’s fixed temporal input dimension posing a constraint on the effective motion periods (ranging from 0.2 to 2.33 seconds). Dealing with motion periods

Table 4.2. Comparison with the state-of-the-art on repetition counting for the *YTSegments* and our *QUVA Repetition* dataset. The deep learning-based method of [Levy and Wolf, 2015](#) achieves good results on their own dataset of relatively clean videos. On our more realistic and challenging dataset, the current method improves considerably over the existing approaches. In comparison to our previous work, our method segments the repetitive motion directly rather than relying on decoupled motion segmentation.

	<i>YTSegments</i>		<i>QUVA Repetition</i>	
	MAE ↓	OBOA ↑	MAE ↓	OBOA ↑
Pogalin et al., 2008	21.9 ±30.1	0.68	38.5 ±37.6	0.49
Levy and Wolf, 2015	6.5 ± 9.2	0.90	48.2 ±61.5	0.45
Our method (Chapter 3)	10.3 ±19.8	0.89	23.2 ±34.4	0.62
Our method	9.4 ±17.4	0.89	26.1 ±39.6	0.62

outside of this range most likely requires retraining the network. The Fourier-based method of [Pogalin et al., 2008](#) scores an MAE of 38.5, whereas we obtain an average error of 26.1. On the *YTSegments* dataset our simplified method slightly improves over the MAE of 10.3 ± 19.8 reported in [Chapter 3](#), while giving comparable results to previously reported MAE of 23.2 ± 34.4 on the *QUVA Repetition* dataset. The Fourier-based and deep learning-based approaches are unable to effectively handle the increased non-stationarity and motion complexity found in our challenging video dataset. The method proposed here improves the ability to handle such difficult videos without relying on explicit motion segmentation methods.

We also report the repetition count results using TV-L¹ ([Zach et al., 2007](#)) and EpicFlow ([Revaud et al., 2015](#)) to investigate our method’s sensitivity to optical flow quality. The results in [Table 4.3](#) show the robustness to different flow methods as the algorithm of choice has limited effect on the count performance for both datasets.

To gain a better understanding of our method’s characteristics we study success and failure cases. We observe that our wavelet-based motion segmentation struggles with scenes containing dynamic textures such as sand or water (e.g. [Figure 3.6](#); bottom row). Based on our analysis, we believe the reason for this is two-fold: (1) For such regions, motion estimation using optical flow is difficult ([Adelson, 2001](#)); and (2) Dy-

Table 4.3. Sensitivity of our method with respect to different optical flow methods. We report repetition counting results over both datasets. Only slight variation in the performance is observed, demonstrating our method’s robustness to optical flow quality.

	<i>YTSegments</i>		<i>QUVA Repetition</i>	
	MAE ↓	OBOA ↑	MAE ↓	OBOA ↑
TV-L ¹	9.8 ±17.9	0.89	26.5 ±67.5	0.67
EpicFlow	9.7 ±17.9	0.88	30.8 ±38.2	0.55
FlowNet 2.0	9.4 ±17.4	0.89	26.1 ±39.6	0.62

dynamic textures produce visual repetitive dynamics resulting in a strong wavelet response over its entire surface. Consequently, motion segmentation by mean-thresholding of the spectral power will fail inevitably; and subsequent measurements over the foreground motion mask will be incorrect as well. For such videos, we observe an enormous over-count as the frequency estimates correspond to the high-frequent rippling water. The error associated with these videos explains the limited improvement over our previous method (Chapter 3) which relied on Papazoglou and Ferrari, 2013 for motion segmentation, being less prone to such segmentation failures. To remedy the problem of coarse and inaccurate segmentation masks, a post-processing step (e.g. conditional random field) is likely to improve the overall segmentation quality.

4.6 Conclusion

In this chapter, we have reconsidered the practical challenges associated with repetition estimation such as the wide variety in motion appearance, non-stationary temporal dynamics and camera motion. Our improved method addresses all these challenges by computing a diversified motion representation, employing the continuous wavelet transform and combining the power spectra of all representations to support viewpoint invariance. Whereas related work explicitly localizes the foreground motion, our method performs repetitive motion segmentation directly from the wavelet power maps resulting in a simplified approach. We verify our claims by improving the state-of-the-art on the task of repetition counting on our challenging new video dataset. Unlike well-performing

existing work, our method requires no learning and requires only a minimum number of hyper-parameters which are fixed throughout our experiments. We envision applications beyond repetition estimation as the wavelet power and scale maps can support localization of low- and high-frequency regions suitable for region pruning or action classification in video (Cheng et al., 2004).

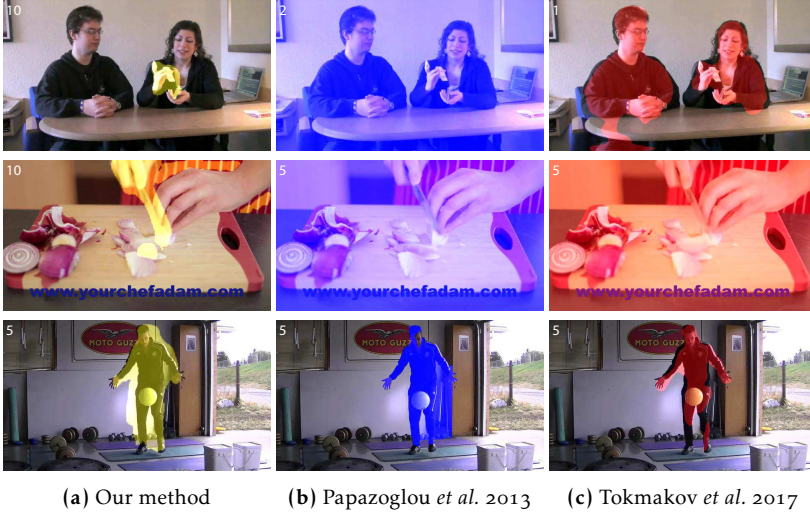


Figure 4.9. Visualization of localization failure cases on the YTSegments dataset. The count prediction is indicated in the top-left corner. From top-to-bottom, the ground-truth counts are 10, 10, 11. *Top row:* our method accurately segments the repetitive foreground motion whereas the other methods fail. As a consequence, we achieve a perfect count whereas the others do not. *Center row:* Again, the wavelet-based localization extracts the repetitive foreground motion, yielding an accurate count result. *Bottom row:* Whereas all methods localize the person with the football, the foreground mask is too coarse.

PART II

RECURRENT PHYSICAL DYNAMICS

*“In the beginner’s mind there are many possibilities,
but in the expert’s mind there are few.” — Shunryū Suzuki*

PART 2 of this thesis is dedicated to the notion of visual recurrence for learning from video of physical phenomena. Chapter 5 will focus on the recurrent motion of cloth in the wind to learn physical parameters from visual observations. In Chapter 6, we will consider recurrent physical dynamics as training data for learning physical properties and relationships from observations only.

CHAPTER 5



CLOTH IN THE WIND

5.1 Introduction

There is substantial evidence (Craig, 1967; Hegarty, 2004) that humans run mental models to predict physical phenomena. We predict the trajectory of objects in mid-air, estimate a liquid’s viscosity and gauge the velocity at which an object slides down a ramp. In analogy, simulation models usually optimize their parameters by performing trial runs and selecting the best. Over the years, physical models of the world have become so visually appealing through simulations and rendering (H. Wang et al., 2011; Narain et al., 2012; Bridson et al., 2005; Schreck et al., 2019) that it is worthwhile to consider them for physical scene understanding. This alleviates the need for meticulous annotation of the pose, illumination, texture and scene dynamics as the model delivers them for free.

In this chapter, we consider *flags* and *cloth* in the wind as a case study of real-world recurrent motion. Measurements and visual models of flags and cloth are important for virtual clothing try-on (Yang et al., 2018), energy harvesting and biological systems (Shelley and Zhang, 2011; Huang and Sung, 2010). The cloth’s intrinsic material properties, together with the external wind force, determine its dynamics. Untangling the dynamics of fabric is challenging due to the involved nature of the air-cloth interaction: a flag exerts inertial and elastic forces on the surrounding air, while the air acts on the fabric through pressure and viscosity (Huang and Sung, 2010). As we seek to measure both the cloth’s intrinsic material properties and the external wind force, our physical

This chapter is based on our CVPR 2020 publication (Runia et al., 2020a).

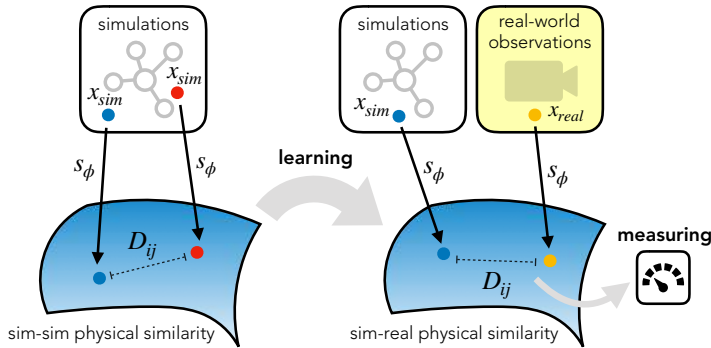


Figure 5.1. We propose to measure real-world physical cloth parameters without ever having seen the phenomena before. From cloth simulations only, we learn a distance metric that encodes both intrinsic and extrinsic physical properties. After learning, we use the embedding function to measure physical parameters from real-world video by comparison to its simulated counterpart.

model couples the non-linear cloth model from [H. Wang et al., 2011](#) with external wind force of [Wejchert and Haumann, 1991](#).

The task is challenging, as physical models of cloth tend to have high numbers of unknown parameters and bear intricate coupling of intrinsic and external forces. Our solution is to compare pairs of real and simulated observations and measure their physical similarity. As there is a fundamental caveat in the use of simulation and rendering for learning: “visually appealing” does not necessarily imply the result is realistic, the main question is how to assess the similarity of the causally underlying physical parameters rather than visual correspondence. It might be the case that the image looks real but never occurs in reality.

At the core of our measurement is a cloth simulation engine with unknown parameters θ to be determined. The outcome of a simulation (e.g. 3D meshes, points clouds, flow vectors) is converted to the image space using a render engine. We then compare the simulated visual data with a real-world observation of the particular phenomenon ([Figure 5.1](#)). Accordingly, we propose to learn a *physical similarity* metric from simulations only, without ever perceiving a real-world example. In the learned embedding space, observations with similar physical parameters will wind up close, while dissimilar example pairs will be further away. Guided by the physical similarity, the simulation’s parameters are

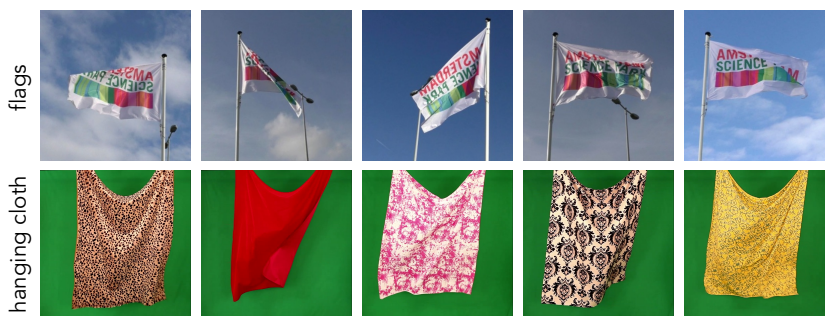


Figure 5.2. We consider two cases of cloth in the wind. Top row: random still images from our video recordings of real flags. For review purposes we have obfuscated text on the flag. Bottom row: examples from the hanging cloth dataset from Bouman et al., 2013.

refined in each step. As a result, we obtain a complete computational solution for the refined measurements of physical parameters.

The contributions of this chapter are as follows: (1) We propose to train a perception-based physical cloth measurement device from simulations only, without ever observing a real-world manifestation of the phenomena. Our measurement device is formulated as a comparison between two visual observations implemented as a Siamese network that we train with contrastive loss. (2) In a case study of cloth, we propose a specific instantiation of the physical embedding function. At its core is a new spectral decomposition layer that measures the spectral power over the cloth’s surface. Our solution compares favorably to existing work that recovers intrinsic and extrinsic physical properties from visual observations. (3) To evaluate our method, we record real-world video of flags with the ground-truth wind speed gauged using an anemometer. (4) Finally, we iteratively refine physics simulations from a single real-world observation towards maximizing the physical similarity between the real-world and its simulation.

5.2 Related Work

Previous work has measured physical properties by perceiving real-world objects or phenomena — including material properties (Bouman et al., 2013), cloth stiffness and bending parameters (Bouman et al., 2013; Yang et al., 2017), mechanical features (J. Wu et al., 2015; Mottaghi et al.,

2016a; Mottaghi et al., 2016b; W. Li et al., 2016), fluid characteristics (J. Wu et al., 2016; Spencer and Shah, 2004; Sakaino, 2008) and surface properties (Meka et al., 2018). The primary focus of the existing literature has been on estimating intrinsic material properties from visual input. However, physical phenomena are often described by the interaction between intrinsic and extrinsic properties. Therefore, we consider the more complex scenario of jointly estimating intrinsic material properties and extrinsic forces from a single real-world video through the iterative refinement of physics simulations.

Our case study focuses on the physics of cloth and flags, both of which belong to the broader category of wind-excited bodies. The visual manifestation of wind has received modest attention in computer vision, *e.g.* the oscillation of tree branches (Xue et al., 2018; Sun et al., 2003), water surfaces (Spencer and Shah, 2004), and hanging cloth (Bouman et al., 2013; Yang et al., 2017; T. Y. Wang et al., 2019; Cardona et al., 2019). Our leading example of a flag curling in the wind may appear simple at first, but its motion is highly complex. Its dynamics are an important and well-studied topic in the field of fluid-body interactions (Shelley and Zhang, 2011; Taneda, 1968; Tian, 2013). Inspired by this work and existing visual cloth representations that characterize wrinkles, folds and silhouette (Bhat et al., 2003; Haddon and Forsyth, 1998; White et al., 2007; Yang et al., 2018), we propose a novel spectral decomposition layer which encodes the frequency distribution over the cloth’s surface.

Previous work has considered the task of measuring intrinsic cloth parameters (Bhat et al., 2003; Bouman et al., 2013; Yang et al., 2017) or external forces (Cardona et al., 2019) from images or video. Notably, Bouman et al., 2013 use complex steerable pyramids to describe hanging cloth in a video, while both Yang et al., 2017 and Cardona et al., 2019 propose a learning-based approach by combining a convolutional network and recurrent network. In our experiments we will compare our cloth frequency-based representations with Cardona et al., 2019 on flags while Yang et al., 2017 is a reference on the hanging cloth dataset.

Our approach of measuring physical parameters by iterative refinement of simulations shares similarity to the Monte Carlo-based parameter optimization of J. Wu et al., 2015 and the particle swarm refinement of clothing parameters from static images (Yang et al., 2018). In particular, the work of Yang et al., 2018 resembles ours as they infer garment properties from images for the purpose of virtual clothing try-on. However, our work is different in an important aspect: we estimate intrinsic

and extrinsic physical parameters from video while their work focuses on estimating intrinsic cloth properties from static equilibrium images. Recently, [Liang et al., 2019](#) have proposed a differentiable cloth simulator which could potentially be used as an alternative to our approach for cloth parameter estimation.

5.3 Method

We consider the scenario in which we make an observation of some phenomena with a physical model explaining its manifestation available to us. Based on the perception of reality, our goal is to measure the D_p unknown continuous parameters of the physical model $\theta \in \mathbb{R}^{D_p}$, consisting of intrinsic parameters θ_i and extrinsic parameters θ_e through an iterative refinement of a computer simulation that implements the physical phenomena at hand. In particular, we consider observations in the form of short video clips $\mathbf{x}_{\text{target}} \in \mathbb{R}^{C \times N_t \times H \times W}$, with C denoting the number of image channels and N_t the number of $H \times W$ frames. In each iteration, the simulator runs with current model parameters θ to produce some intermediate representation (*e.g.* 3D meshes, point clouds or flow vectors), succeeded by a render engine with parameters ζ that yields a simulated video $\mathbf{x}_{\text{sim}} \in \mathbb{R}^{C \times N_t \times H \times W}$. Our insight is that the physical similarity between real-world observation and simulation can be measured in some embedding space using pairwise distance:

$$D_{i,j} = D(s_\phi(\mathbf{x}_i), s_\phi(\mathbf{x}_j)) : \mathbb{R}^{D_e} \times \mathbb{R}^{D_e} \rightarrow \mathbb{R}, \quad (5.1)$$

with the embedding function:

$$s_\phi(\mathbf{x}) : \mathbb{R}^{C \times N_t \times H \times W} \rightarrow \mathbb{R}^{D_e}. \quad (5.2)$$

The embedding function maps the data manifold $\mathbb{R}^{C \times N_t \times H \times W}$ to some embedding manifold \mathbb{R}^{D_e} using a neural network parametrized by ϕ such that physically similar examples lie close. In each iteration, guided by the pairwise distance (5.1) between real and simulated instance, the physical model is refined to maximize physical similarity. This procedure ends whenever the physical model parameters have been measured accurately enough or when the evaluation budget is finished. The output comprises the measured physical parameters θ^* and corresponding

simulation $\mathbf{x}_{\text{sim}}^*$ of the real-world phenomenon. An overview of the proposed method is presented in Figure 5.3.

5.3.1 Physical Similarity

For the measurement to be successful, it is crucial to measure the similarity between simulation \mathbf{x}_{sim} and real-world observation $\mathbf{x}_{\text{target}}$. The similarity function must reflect correspondence in physical dynamics between the two instances. The prerequisite is that the physical model must describe the phenomenon’s behavior at the scale that coincides with the observational scale. For example, the quantum mechanical understanding of a pendulum will be less meaningful than its formulation in classical mechanics when capturing its appearance using a regular video camera.

Given the physical model and its implementation as a simulation engine, we generate a dataset of simulations with its parameters θ randomly sampled from some predefined search space. For each of these simulated representations of the physical phenomenon, we use a 3D render engine to generate multiple video clips $\mathbf{x}_{\text{sim}}^i$, with different render parameters ζ^i . As a result, we obtain a dataset with multiple renders for each simulation instance. Given this dataset we propose the following training strategy to learn a distance metric quantifying the physical similarity between observations.

We employ a *contrastive loss* (Hadsell et al., 2006; LeCun et al., 2006) and consider positive example pairs to be rendered video clips originating from the same simulation (*i.e.* sharing physical parameters) while negative example pairs have different physical parameters. Both rendered video clips of an example pair are mapped to the embedding space through $s_\phi(\mathbf{x})$ in Siamese fashion (Bromley et al., 1994). In the embedding space, the physical similarity will be evaluated using the squared Euclidean distance:

$$D_{i,j} = D(s_\phi(\mathbf{x}_i), s_\phi(\mathbf{x}_j)) = \|s_\phi(\mathbf{x}_i) - s_\phi(\mathbf{x}_j)\|_2^2. \quad (5.3)$$

If optimized over a collection of rendered video clips, the contrastive loss will learn to pull physically similar examples together, whereas physically dissimilar points will be pushed apart. As a result, by training on simulations only, we can learn to measure the similarity between simulations and the real-world pairs.

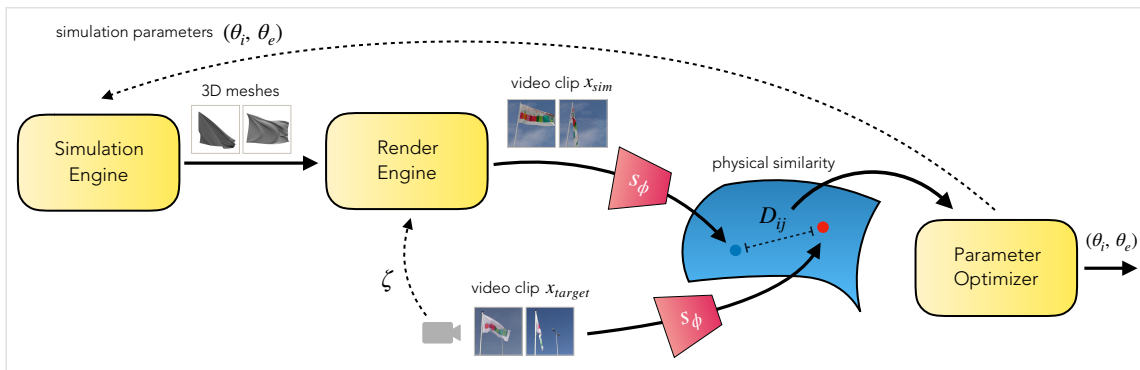


Figure 5.3. In this chapter, we propose the perception-based measurement of physical scene properties. Given an observation of a real-world physical phenomenon, here represented as video clip $\mathbf{x}_{\text{target}}$, our algorithm measures the underlying parameters of the physical scene. Central is a simulation engine implementing the physical model, parametrized by intrinsic material properties θ_i and the characterization of external forces θ_e . A render engine, with render parameters ζ , maps the simulator's output to the image space producing video clip \mathbf{x}_{sim} . Using an embedding function $s_\phi(\mathbf{x})$, both real and simulated examples are mapped to a manifold on which physically similar examples are assigned to nearby points. To measure the similarity between both clips, we evaluate a distance metric $D_{ij}(\cdot, \cdot)$ in the embedding space. Its result serves as the objective for an optimization module that refines the physical parameters θ towards the actual observation.

5.3.2 Simulation Parameter Optimization

We propose to obtain a physical measurements through gradual refinement of simulations based on visual observations. See [Figure 5.3](#) for an overview of the proposed refinement pipeline.

To optimize the physical parameters we draw the parallel with the problem of hyper-parameter optimization ([Snoek et al., 2012](#); [Bergstra and Bengio, 2012](#)). In light of this correspondence, our collection of model parameters is analogous to the hyper-parameters involved by training deep neural networks (e.g. learning rate, weight decay, dropout). Formally, we seek to find the global optimum of physical parameters:

$$\theta^* = \operatorname{argmin}_{\theta} D(s_{\phi}(\mathbf{x}_{\text{target}}), s_{\phi}(\mathbf{x}_{\text{sim}}(\theta))), \quad (5.4)$$

where the target example is fixed and the simulated example depends on the current set of physical parameters θ . Adjusting the parameters θ at each iteration is challenging as it is hard to make parametric assumptions on (5.4) as function of θ and accessing the gradient is costly due to the simulations' computational complexity. Our goal is, therefore, to estimate the global minimum with as few evaluations as possible. Considering this, we adopt Bayesian optimization ([Snoek et al., 2012](#)) for updating parameters θ . Its philosophy is to leverage all available information from previous observations of (5.4) and not only use local gradient information. We treat the optimization as-is and use a modified implementation of Spearmint from [Snoek et al., 2012](#) with the Matérn52 kernel and improved initialization of the acquisition function proposed by [Oh et al., 2018](#). We emphasize that, at this stage, the embedding function $s_{\phi}(\mathbf{x})$ is already trained and remains fixed throughout the optimization procedure.

5.3.3 Physics, Simulation and Appearance of Cloth

Up until now, we have discussed the proposed method in its most general terms. Here, we will transition to a specific instantiation of a physical phenomenon. Specifically, we will consider two cases of cloth exposed in the wind: curling flags and fabric hanging on a rod ([Figure 5.4](#)). By doing so, we confine the parameters θ and embedding function $s_{\phi}(\mathbf{x})$. However, we could have considered other physical phenomena given

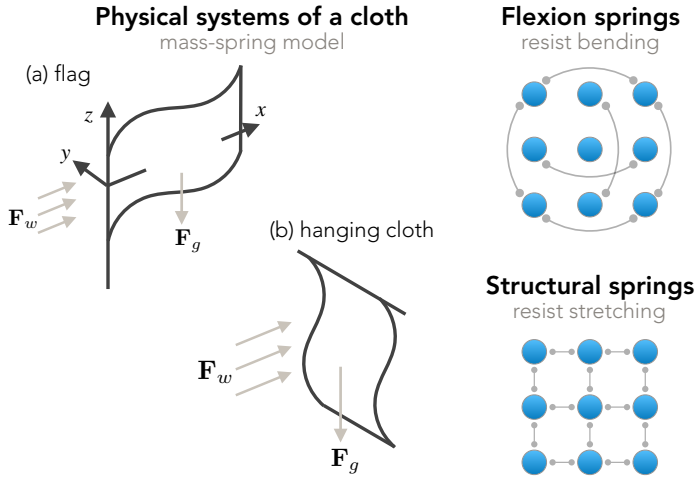


Figure 5.4. *Left:* we consider two cases of cloth exposed in the wind: (a) a flag curling in the wind; and (b) cloth fabric hanging from a rod. In both cases, the fabric fabric is treated as a mass-spring model in which a dense grid of point masses is inter-connected with multiple springs. *Right:* the bending and stretching springs determine the materials behavior. Flexion springs act over shared edges whereas structural springs connect to direct neighbors.

that the underlying physical model is known, and a perceptual distance function can be learned.

5.3.4 Physical Model

Researchers have been fascinated by flags' seemingly simple, yet highly complex, behavior that arises from the cloth-wind interaction (Taneda, 1968; J. Zhang et al., 2000; Huang and Sung, 2010; Eloy et al., 2008; Shelley and Zhang, 2011; Tian, 2013). This complex coupling can give rise to the flag's familiar appearance of self-sustained oscillation at certain natural frequencies. Furthermore, the physical understanding of cloth and its interaction with external forces has been assimilated by the computer graphics community. Most successful methods treat cloth as a mass-spring model: a dense grid of point masses organized in a planar structure, inter-connected with different types of springs which properties determine the fabric's behavior (Baraff and Witkin, 1998; Provot et al., 1995; H. Wang et al., 2011; Narain et al., 2012). We adopt the non-linear and anisotropic mass-spring cloth model introduced by

H. Wang et al., 2011. This model uses a piecewise linear bending and stretching models. The stretching model is a generalization of Hooke's law for continuous media (Slaughter, 2012). As our experiments focus on flags in the wind for which the stretching properties are of minimal relevance, our experiments will focus on flags in the wind, typically made of strong weather-resistant material such as polyester and nylon. Therefore, the material's stretching properties are of minimal relevance and we will focus on the bending model from H. Wang et al., 2011 and external force model of Wejchert and Haumann, 1991.

Bending Model

The bending model is based on the linear bending force equation first proposed in Bridson et al., 2005. The model formulates the elastic bending force \mathbf{F}_e over triangular meshes sharing an edge (Figure 5.4). For two triangles separated by the dihedral angle φ , the bending force takes the form of:

$$\mathbf{F}_e = k_e \sin(\varphi/2)(N_1 + N_2)^{-1}|\mathbf{E}|\mathbf{u}, \quad (5.5)$$

where k_e is the material dependent bending stiffness, N_1, N_2 are the weighted surface normals of the two triangles, \mathbf{E} represents the edge vector and \mathbf{u} is the bending mode (see Figure 1 in Bridson et al., 2005). The bending stiffness k_e is non-linearly related to the dihedral angle φ . This is realized by treating k^e as piecewise linear function of the reparametrization $\alpha = \sin(\varphi/2)(N_1 + N_2)^{-1}$. After this reparametrization, for a certain fabric, the parameter space is sampled for N_b angles yielding a total of $3N_b$ parameters across the three directions. H. Wang et al., 2011 empirically found that 5 measurements are sufficient for most fabrics, producing 15 bending parameters.

External Forces

For the dynamics of cloth, we consider two external forces acting upon its planar surface. First, the Earth's gravitational acceleration ($\mathbf{F}_g = m\mathbf{a}_g$) naturally pushes down the fabric. The total mass is defined by the cloth's area weight ρ_A multiplied by surface area. More interestingly, we consider the fabric exposed to a constant wind field. Again, modeling the cloth as a grid of point masses, the drag force on each mass is stipulated by Stokes's equation $\mathbf{F}_d = 6\pi R\eta\mathbf{v}_w$ in terms of the surface area, the air's

dynamic viscosity and the wind velocity \mathbf{v}_w (Wejchert and Haumann, 1991; Narain et al., 2012). By all means, this is a simplification of reality. Our model ignores terms associated with the Reynolds number (such as the cloth’s drag coefficient), which will also affect a real cloth’s dynamics. However, it appears that the model is accurate enough to cover the spectrum of cloth dynamics.

5.3.5 Simulation Engine

We employ the non-differentiable ArcSim simulation engine (Narain et al., 2012) which efficiently implements the complex physical model described in Section 5.3.4. On top of the physical model, the simulator incorporates anisotropic remeshing to improve detail in densely wrinkled regions while coarsening flat regions. As input, the simulator expects the cloth’s initial mesh, its material properties and the configuration of external forces. At each time step, the engine solves the system for implicit time integration using a sparse Cholesky-based solver. This produces a sequence of 3D cloth meshes based on the physical properties of the scene. As our goal is to learn a physical distance metric in image space between simulation and a real-world observation, we pass the sequence of meshes through a 3D render engine (Blender, 2018). Given render parameters ζ comprising of camera position, scene geometry, lighting conditions and the cloth’s visual texture, the renderer produces a simulated video clip (\mathbf{x}_{sim}) which we can compare directly to the real-world observation ($\mathbf{x}_{\text{target}}$). We emphasize that our focus is neither on inferring render parameters ζ from observations nor on attaining visual realism for our renders.

Parameter Search Space

The ArcSim simulator (Narain et al., 2012) operates in metric units, enabling convenient comparison with real-world dynamics. As the base material for our flag experiments, we use “Camel Ponte Roma” from H. Wang et al., 2011. Made of 60% polyester and 40% nylon, this material closely resembles widely used flag fabrics. The fabric’s bending coefficients, stretching coefficients, and area weight were accurately measured in a mechanical setup by the authors. We adopt and fix their stretching parameters and use the bending stiffness and area weight as initialization for our cloth material. Specifically, using their respective parameters

Table 5.1. The predefined parameter range for optimization of $\theta = (\theta_i, \theta_e)$ given the physical model of a flag curling in the wind. The bending parameters \bar{k}_e correspond to the “Camel Ponte Roma” material from [H. Wang et al., 2011](#).

	Parameter	Params	Search space
θ_i	Bending stiffness	15	$k_e \in [10^{-1}\bar{k}_e, 10\bar{k}_e]$
θ_i	Fabric area weight	1	$\rho_A \in [0.10, 0.17] \text{ kg m}^{-2}$
θ_e	Wind velocity	1	$v_w \in [0, 10] \text{ m s}^{-1}$

we confine a search space that is used during our parameter refinement. We determine $\rho_A \sim \text{Uniform}(0.10, 0.17) \text{ kg m}^{-2}$ after consulting various flag materials at online retailers. And, we restrict the range of the bending stiffness coefficients by multiplying the base material’s \bar{k}_e in (5.5) by 10^{-1} and 10 to obtain the most flexible and stiffest material respectively. As the bending coefficients have a complex effect on the cloth’s appearance, we independently optimize the 15 bending coefficients instead of only tuning the one-dimensional multiplier. The full parameter search space is listed in [Table 5.1](#).

5.3.6 Spectral Decomposition Network

As evident from [Section 5.3.4](#), the behavior of flags is highly complex on a microscopic level. However, as we will demonstrate, we can infer both intrinsic and extrinsic properties of the physical system from its visual appearance at a typical observational scale. The manifestation of a flag will express various distinct cues that can be leveraged for deducing its physical parameters. In the absence of wind, a flag will be hanging down. As the wind catches, the flag will start curling. Gradually, waves will start traveling towards its trailing edge. At certain speeds, the flag will exhibit self-sustained periodic motion ([Chapter 2](#)) at its natural frequency. Picking up, the wind’s lashing force can induce violent flapping and turbulent behavior. Over the wind speed spectrum, generally, the flag’s lower side exhibits high-frequent traveling waves towards its trailing edge while the upper side is more stable and dominantly displays low-frequent vertical motion.

The dominant source of variation is in the geometry of the waves in cloth rather than in its texture. Therefore, we seek a perceptual model

that can encode the cloth’s dynamics such as high-frequent streamwise waves, the number of nodes in the fabric, violent flapping at the trailing edge, rolling motion of the corners and its silhouette (Shelley and Zhang, 2011; Taneda, 1968; Eloy et al., 2008). As our goal is to measure sim-to-sim and sim-to-real similarity, a crucial underpinning is that our embedding function is able to disentangle and extract the relevant signal for domain adaptation (Peng et al., 2018; James et al., 2019). Therefore, building upon Chapter 4, we propose modeling the *spatial distribution of temporal spectral power* over the cloth’s surface. Together with direction awareness, this effectively characterizes the traveling waves and flapping behavior from visual observations.

Spectral Decomposition Layer

The proposed solution is a novel spectral decomposition layer that extracts temporal frequencies from a video. Specifically, similar to our method for repetition estimation as described in Chapter 4, we treat an input video volume as a collection of signals for each spatial position (*i.e.* $H \times W$ signals) and map the signals into the frequency domain using the Discrete Fourier Transform (DFT) to estimate the videos’ spatial distribution of temporal spectral power. The DFT maps a signal $f[n]$ for $n \in [0, N_t - 1]$ into the frequency domain (Oppenheim, 1999):

$$F(j\omega) = \sum_{n=0}^{N_t-1} f[n]e^{-j\omega nT}. \quad (5.6)$$

We proceed by mapping the Fourier transform’s complex output to a real-valued representation. The periodogram of a signal is a representation of its spectral power and is defined as:

$$I(\omega) = \frac{1}{N_t} |F(j\omega)|^2, \quad (5.7)$$

with $F(j\omega)$ as defined in (5.6). This provides the spectral power magnitude at each sampled frequency. To effectively reduce the dimensionality and emphasize on the videos’ discriminative frequencies, we select the top- k strongest frequencies and corresponding spectral power from the periodogram. Given a signal of arbitrary length, this produces k pairs containing the spectral power $I(\omega_{\max_i})$ and frequency ω_{\max_i} for $i \in [0, k]$ yielding a total of $2k$ scalar values.

Algorithm 1 Spectral Decomposition Layer

```
1: Input. Video tensor  $\mathbf{x}$  of shape  $[N_b, C, N_t, H, W]$ 
2: Input. Number of frequency peaks to select,  $k$ 
3: Output. Decomposition of shape  $[N_b, 2kC, H, W]$ 

4: procedure SPECTRALDECOMPOSITIONLAYER( $\mathbf{x}$ )
5:   Reshape  $\mathbf{x}$  to  $[N_b C H W, N_t]$  to obtain batch of signals
6:   Apply a Hanning window to signals
7:   Compute the DFT of signals using (5.6)
8:   Compute periodogram of signals  $I(\omega)$ 
9:   Select top- $k$  peaks of  $I(\omega)$  and corresponding  $\omega$ 's
10:   $P \leftarrow$  top- $k$  peaks of  $I(\omega)$  reshaped to  $[N_b, kC, H, W]$ 
11:   $\Omega \leftarrow$  corresponding  $\omega$ 's reshaped to  $[N_b, kC, H, W]$ 
12:  return  $P, \Omega$ 
13: end procedure
```

Considering an input video volume, treated as a collection of $H \times W$ signals of length N_t , the procedure extracts the discriminative frequency and its corresponding power at each spatial position. In other words, the spectral decomposition layer performs the mapping:

$$\mathbb{R}^{C \times N_t \times H \times W} \rightarrow \mathbb{R}^{2kC \times H \times W}. \quad (5.8)$$

The videos' temporal dimension is squeezed and the result can be considered a multi-channel feature map. Consequently, these can be further processed by any 2D convolutional layer. We reduce spectral leakage using a Hanning window before applying the DFT. In [Algorithm 1](#), we formalize the optimized batched version of the spectral decomposition layer as we implement for our experiments.

Embedding Function

The specification of $s_\phi(\mathbf{x})$, with the spectral decomposition layer at its core, is illustrated in [Figure 5.5](#). First, our model convolves the input video \mathbf{x} with a temporal Gaussian filter followed by two spatially oriented first-order derivative filters. Both resulting video volumes are two-times spatially subsampled by means of max-pooling. Successively, the filtered video representations are fed through the spectral decomposition layer to produce spectral power and frequency maps. The outputs are stacked into a multi-channel feature map to be further processed

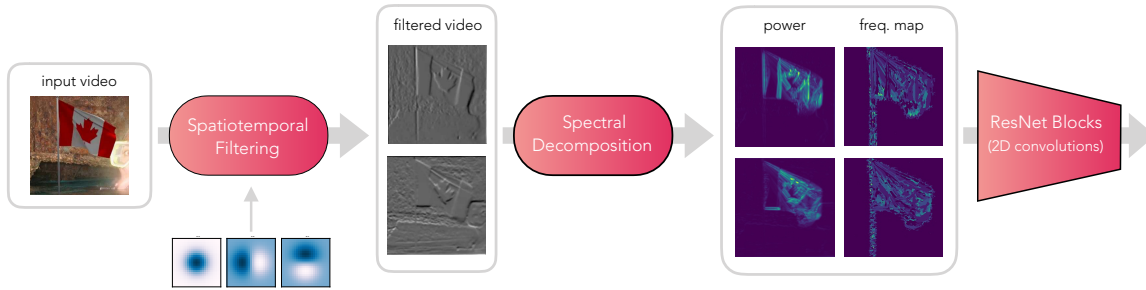


Figure 5.5. Overview of our SDN architecture $s_\phi(\mathbf{x})$ for learning the physical correspondence between the simulation and real-world observation of dynamic flags. Given a 3D video volume as input, we first apply a 0th-order temporal Gaussian filter followed by two directional 1st-order Gaussian derivative filters and then spatially subsample both filtered video volumes by a factor two. The proposed spectral decomposition layer then applies the Fourier transform and selects the maximum power and corresponding frequencies densely for all spatial locations. This produces 2D multi-channel feature maps which we process with 2D ResNet blocks to learn the embedding.

by a number of 2D convolutional filters with trainable weights ϕ . We use 3 standard ResNet blocks [He et al., 2016](#) and a final linear layer that maps to the \mathbb{R}^{D_e} embedding space. We refer to our network as *Spectral Decomposition Network (SDN)*.

Network Details

Our spectral decomposition network is implemented in PyTorch ([Paszke et al., 2019](#)). Unless mentioned otherwise, all network inputs are temporally sampled at 25 fps. After that, we use a temporal Gaussian with $\sigma_t = 1$ and first-order Gaussian derivative filters with $\sigma_{x,y} = 2$. For training the embedding function with the contrastive loss, we adopt a margin of 1 and use the *BatchAll* sampling strategy ([Hermans et al., 2017](#); [Ding et al., 2015](#)). The spectral decomposition layer selects the single most discriminative frequency (*i.e.* $k = 1$). Adding secondary frequency peaks to the feature maps did not yield substantial performance gains. The size of our embeddings is fixed ($D_e = 512$) throughout the chapter. Input video clips of size 224×224 are converted to grayscale. We optimize the weights of the trainable ResNet blocks using Adam ([Kingma and Ba, 2015](#)) with mini-batches of 32 examples and an initial learning rate of 10^{-2} . To prevent overfitting, we utilize weight decay of $2 \cdot 10^{-3}$ for all networks. Training continues until validation loss plateaus — typically around 40 epochs. Total training time for our spectral decomposition network is about 4 hours on a single Nvidia GeForce GTX Titan X. When training the recurrent models ([Cardona et al., 2019](#); [Yang et al., 2017](#)) we also perform gradient clipping (max norm of 10) to improve stability.

5.4 Real and Simulated Datasets

We here describe the three datasets used in this chapter. First we describe our flag dataset consisting of real-world videos, after which we transition to the discussion of our two simulation datasets.

5.4.1 Real-world Flag Videos

To evaluate our method’s ability to infer physical parameters from real-world observations, we have set out to collect video recordings of real-world flags with ground-truth wind speed. We used two anemometers



Figure 5.6. *Left:* Two anemometers used for gauging the wind speed. *Right top:* Real flag recordings with corresponding wind speeds measured by the anemometer hoisted in the flagpole. *Right bottom:* simulated examples from our FlagSim dataset.

(Figure 5.6) to measure the wind speed at the flag’s position. After calibration and verification of the meters, we hoisted one of them in the flagpole to the height of the flag to ensure accurate and local measurements. A Panasonic HC-V770 camera was used for video recording. In total, we have acquired more than an hour of video over the course of 5 days in varying wind and weather conditions. We divide the dataset in 2.7K train and 1.3K non-overlapping test video clips and use 1-minute average wind speeds as ground-truth. The train and test video clips are recorded on different days with varying weather conditions. Examples of the videos are displayed in Figure 5.6.

Data Acquisition

We here elaborate on our data acquisition to obtain real-world wind speed measurements serving as ground-truth for the final experiment of this chapter. To accurately gauge the wind speed next to the flag, we have obtained two anemometers:

- SkyWatch BL-400: windmill-type anemometer
- Testo 410i: vane-type anemometer

The measurement accuracy of both meters anemometers is 0.2 ms^{-1} . To verify the correctness of both anemometers, we have checked that both wind meters report the same wind speeds before usage. After that, we use the SkyWatch BL-400 anemometer for our measurements as it measures omnidirectional which is more convenient. We hoisted the anemometer in a flag pole such that the wind speeds are measured at the same height as the flag. Wind speed measurements are recorded

at 1 second intervals and interfaced to the computer. In [Figure 5.7](#) we display an example measurement and report the dataset’s wind speed distribution. For the experiments, we randomly sample video clips of 30 consecutive frames from our video recordings and consider the ground-truth wind speed to be the average over the last minute. This procedure ensures that small wind speed deviations and measurement errors are averaged out over time.

The camera records at 1920×1080 at 60 frames per second. We perform post-processing of the videos in the following ways. Firstly, we temporally subsample the video frames at 25 fps such that the clips are in accordance with the frame step size in the physics simulator. Moreover, we assert that the video recordings are temporally aligned with the wind speed measurements using their timestamps. Secondly, we manually crop the videos such that the curling flag appears in the approximate center of the frame. After this, the frames are spatially subsampled to 300×300 , again in agreement with animations obtained from the render engine ([Blender, 2018](#)).

5.4.2 FlagSim Dataset

To train the embedding function $s_\phi(\mathbf{x})$ as discussed in [Section 5.3.1](#), we introduce the FlagSim dataset consisting of flag simulations and their rendered animations. We simulate flags by random sampling a set of physical parameters θ from [Table 5.1](#) and feed them to ArcSim. For each flag simulation, represented as sequence of 3D meshes, we use [Blender, 2018](#) to render multiple flag animations $\mathbf{x}_{\text{sim}}^i$ at different render settings ζ^i . We position the camera at a varying distance from the flagpole and keep a maximum angle of 15° between the wind direction and camera axis. From a collection of 12 countries, we randomly sample a flag texture. Background images are selected from the SUN397 dataset ([Xiao et al., 2010](#)). Each simulation produces 60 cloth meshes at step size $\Delta T = 0.04$ s corresponding to 25 fps, which we render at 300×300 resolution. Following this procedure, we generate 1,000 mesh sequences and render a total of 14,000 training examples. We additionally generate validation and test sets of 150/3,800 and 85/3,500 mesh sequences/renderers respectively. See [Figure 5.6](#) (bottom row) for example renders.

The examples in the FlagSim dataset are written to disk as a sequence of 60 JPEG frames of size 300×300 . During training, when using less than 60 input frames (30 is used in most experiments), we randomly

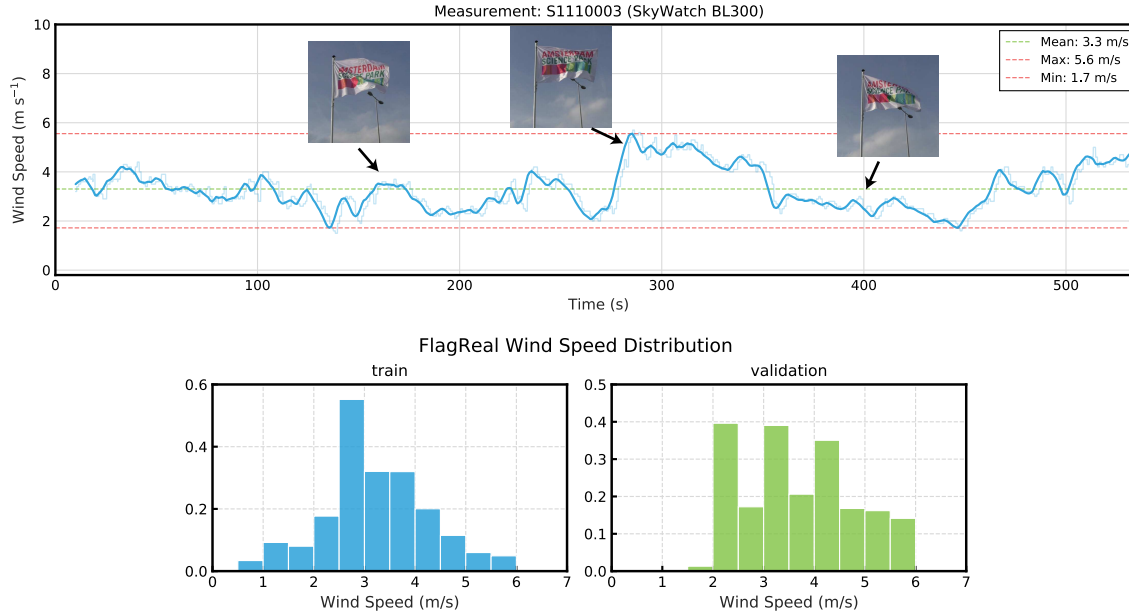


Figure 5.7. *Top:* Example of the time-varying wind speed as obtained by the SkyWatch BL-400 anemometer positioned directly next to the video-recorded flag. The wind speed is sampled at 1 Hz and interfaced to a computer using bluetooth. For our final experiment, we sample video clips of 30 frames and consider the ground-truth wind speed to be the average wind speed over the last minute. *Bottom:* Distribution statistics of the dataset we collected. Over all 4K non-overlapping videos the average wind speed is 3.2m s^{-1} while the minimum and maximum wind speeds are 0.5m s^{-1} and 6.0m s^{-1} respectively.

sample N_t successive frames from each video clip. This is achieved by uniform sampling of a temporal offset within the video. After this, for the sampled sequence of frames, we convert images to grayscale, perform multi-scale random cropping and apply random horizontal flipping (L. Wang et al., 2015) to obtain a $N_t \times 1 \times 224 \times 224$ input clip. Before feeding it into the network, we subtract the mean and divide by the standard deviation for each video clip.

5.4.3 ClothSim Dataset

Our real-world flag dataset enables us to evaluate our method’s measurement performance of external parameters ($v_w \in \theta_e$) by means of the measured wind speed. However, the cloth’s internal parameters are unknown and cannot be evaluated beyond visual inspection. Therefore, we also perform experiments on the hanging cloth dataset of Bouman et al., 2013. The authors have carefully determined the internal cloth material properties, which we can leverage for quantitative evaluation of our simulated-refined measurements. Specifically, we assess our method’s ability to measure the cloth’s *area weight* (kg m^{-2}) in a similar fashion as for cloth. However, we retrain the embedding function $s_\phi(\mathbf{x})$ on a dataset of hanging cloth simulations, referred to as ClothSim.

Following the same procedure as for the FlagSim dataset, we generate a dataset of simulated hanging cloth excited by a constant wind force. The main difference between the FlagSim dataset is the wider variety of cloth material. Specifically, we use all the materials presented in H. Wang et al., 2011 available in ArcSim. The increased diversity allows us to model the dynamics in real-world hanging cloth recording (Bouman et al., 2013). Our dataset shares similarity with the simulated hanging cloth dataset of (Yang et al., 2017). However, they use the dataset for training a ResNet material classifier.

5.5 Results and Discussion

5.5.1 Real-world Extrinsic Wind Speed Measurement

In our first experiment, we assess the effectiveness of the proposed spectral decomposition network by measuring the wind speed on the recently proposed real-world flag dataset by Cardona et al., 2019. Their method, consisting of an ImageNet-pretrained ResNet-18 (He et al.,

2016) with LSTM, will be the main comparison. We also train ResNet-18 with multiple input frames, followed by temporal average pooling of the final activations (Karpathy et al., 2014). After training all methods, we report the root mean squared error (RMSE) and accuracy within 0.5 m s^{-1} (Acc@0.5) in Table 5.2. While our method has significantly fewer parameters (2.6M versus 11.2M and 42.1M), the SDN outperforms the existing work on the task of real-world wind speed regression. This indicates the SDN’s effectiveness in modeling the spatial distribution of spectral power over the cloth’s surface and its descriptiveness for the task at hand. Table 5.3 contains the results on our FlagSim dataset.

Table 5.2. External wind speed prediction from real-world flag observations on the dataset of Cardona et al., 2019. We regress the wind speed ($v_w \in \theta_e$) in the range 0 m s^{-1} to 15.5 m s^{-1} and report numbers on the evaluation split.

Model	Input Modality	RMSE ↓	Acc@0.5 ↑
Cardona et al., 2019	$30 \times 224 \times 224$	1.458	0.301
ResNet-18	$1 \times 224 \times 224$	1.390	0.274
ResNet-18	$10 \times 224 \times 224$	1.237	0.314
ResNet-18	$20 \times 224 \times 224$	1.347	0.296
SDN (ours)	$30 \times 224 \times 224$	1.179	0.337

Table 5.3. External wind speed prediction from simulation. We regress the wind speed ($v_w \in \theta_e$) on our FlagSim dataset. The metrics are computed over the 3.5K test examples. Target velocities range from 0 m s^{-1} (no wind) to 10 m s^{-1} (strong wind). Experimental setup is identical to that of Table 5.2.

Model	Input Modality	RMSE ↓	Acc@0.5 ↑
Yang et al., 2017	$10 \times 227 \times 227$	0.380	0.620
Cardona et al., 2019	$30 \times 227 \times 227$	0.271	0.580
ResNet-18	$1 \times 224 \times 224$	0.381	0.615
ResNet-18	$10 \times 224 \times 224$	0.264	0.734
ResNet-18	$20 \times 224 \times 224$	0.207	0.775
SDN (ours)	$20 \times 224 \times 224$	0.183	0.813
SDN (ours)	$30 \times 224 \times 224$	0.180	0.838

5.5.2 Physical Similarity Quality

We evaluate the physical similarity embeddings after training with contrastive loss. To quantify the ability to separate examples with similar and dissimilar physical parameters, we report the triplet accuracy (Veit et al., 2017). We construct 3.5K FlagSim triplets from the test set as described in Section 5.3.1. We consider the SDN trained for video clips of a varying number of input frames and report its accuracies in Table 5.4. The results indicate the effectiveness of the learned distance metric to quantify the physical similarity between different observations. When considering flags, we conclude that 30 input frames are best with a triplet accuracy of 96.3% and therefore use 30 input frames in the remainder of this chapter. In Figure 5.8 we visualize a subset of the embedding space and observe that the flag instances with low wind speeds are clustered in the top-right corner whereas strong wind speeds live in the bottom-left.

5.5.3 Real-world Intrinsic Cloth Parameter Recovery

In this experiment, we assess the effectiveness of our SDN for estimating intrinsic cloth material properties from a real-world video. We compare our work with Yang et al., 2017 on the hanging cloth dataset of Bouman et al., 2013 (Figure 5.2). Each of the 90 videos shows one of 30 cloth types hanging down while being excited by a fan at 3 wind speeds (W_1 -3). The goal is to infer the cloth’s stiffness and area weight. From our SDN trained on FlagSim with contrastive loss, we extract the embedding vectors for the 90 videos project them into a 50-dimensional space using PCA. Then we train a linear regression model using leave-one-out following Bouman et al., 2013. The results are displayed in Figure 5.9. While not outperforming the specialized method of Yang et al., 2017, we find that our flag-based features generalize to intrinsic cloth material recovery. This is noteworthy, as our SDN was trained on flags of lightweight

Table 5.4. Evaluation of our physical similarity $s_\phi(\mathbf{x})$ for FlagSim test examples. We report average triplet accuracies as suggested by Veit et al., 2017.

Input Frames	10	20	30	40	50
FlagSim accuracy	89.3	92.1	96.3	90.1	92.4
ClothSim accuracy	88.8	93.2	92.5	90.2	91.6



Figure 5.8. Barnes-Hut t-SNE (Van Der Maaten, 2014) visualization of the learned flag embedding space. For visualization purpose we only display examples with wind from the left. Top-right examples exhibit flags at low wind speeds while bottom-left corresponds to strong winds. This indicates that the SDN learns a meaningful representation.

materials exhibiting predominantly horizontal motion. We also test our network’s sensitivity after rotating the hanging cloth videos by 90° counterclockwise to increase visual similarity with flags. However, we obtained comparable results, indicating that our method is fairly robust to the motion direction for estimating intrinsic material properties from real-world video.

5.5.4 Real-world Combined Parameter Refinement

Putting everything together, our goal is measuring physics parameters based on real-world observations. We demonstrate the full measurement procedure (Figure 5.3; method overview) by optimizing over intrinsic and extrinsic model parameters (θ_i, θ_e) from both real-world flag and hanging cloth videos. First, we randomly sample a real-world flag recording as subject of the measurement. The parameter range of the intrinsic

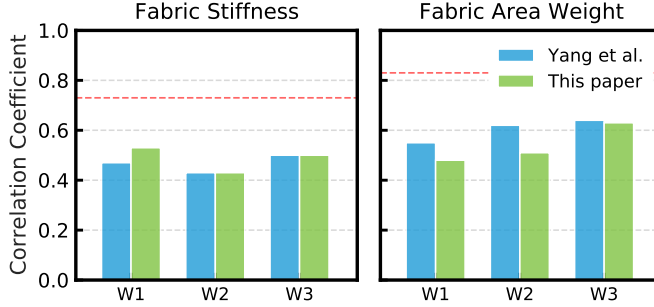


Figure 5.9. Intrinsic cloth material measurements from real videos. We report the Pearson correlation coefficients between predicted material type and both ground-truth stiffness/density on the hanging cloth dataset. The dashed red line indicates human performance as determined by [Bouman et al., 2013](#).

($16\times$) and extrinsic ($1\times$) is normalized to the domain $[-1, +1]$ and are all initialized 0, *i.e.* their center values. As our emphasis is not on inferring the render parameters ζ from the real-world observation, we set its values in oracle fashion, although the embedding function is moderately robust to this variance ([Figure 5.8](#)). In each step, we simulate the cloth meshes with current parameters θ_i, θ_e and render its video clip with fixed render parameters ζ . Both the simulation and real-world video clips are then projected onto the embedding space using $s_\phi(\mathbf{x})$, and we compute their pairwise distance ([5.1](#)). Finally, the Bayesian optimization’s acquisition function ([Section 5.3.2](#)) determines to where make the next evaluation $\theta_i, \theta_e \in [-1, +1]$ to maximize the expected improvement, *i.e.* improving the measurement. The next iteration starts by denormalizing the parameters and running the simulation. We run the algorithm for 50 refinement steps. In [Figure 5.10](#), we demonstrate our method’s measurements throughout optimization. Most importantly, we observe a gradual decrease in the pairwise distance between simulation and real-world example, indicating a successful measurement of the physical parameters. Importantly, we note that the wind speed converges towards the ground-truth wind speed within a few iterations, as indicated with a dashed line. Similar results for refining the intrinsic fabric area weight for the hanging cloth dataset are presented in [Figure 5.11](#).

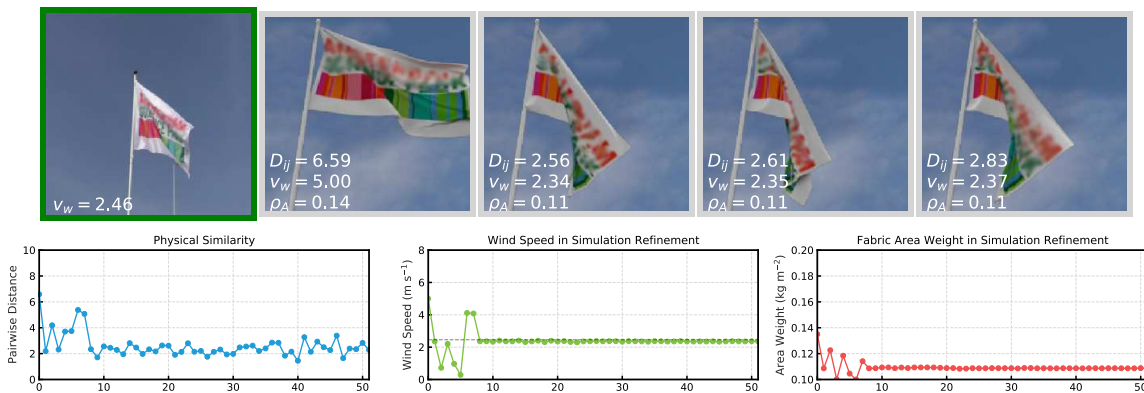


Figure 5.10. Result of our iterative measurement for a target video capturing a flag in the wind. *Top left:* frame from the real-world target video clip with the ground-truth wind speed measured using an anemometer. *Top remaining:* simulated examples throughout the refinement process with corresponding simulation parameters. *Bottom:* development throughout the refinement process for 50 iteration steps. We plot the distance between simulation and target instance in the embedding space and the estimated wind speed (m s^{-1}). We annotate the ground-truth wind speed with a dashed line. As the plot indicates, the refinement process converges towards the real wind speed.

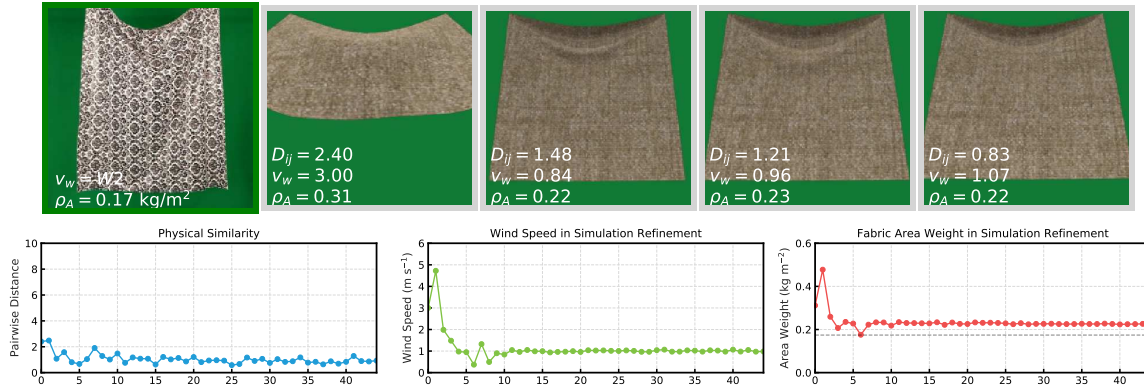


Figure 5.11. Result of our iterative measurement for a target video from the hanging cloth dataset [Bouman et al., 2013](#). *Top left:* frame from the real-world target video clip with the ground-truth fabric area weight measured by [Bouman et al., 2013](#). *Top remaining:* simulated examples throughout the refinement process with corresponding simulation parameters. *Bottom:* development throughout the refinement process for 50 iteration steps. We plot the distance between simulation and target instance in the embedding space, estimated wind speed (m s^{-1}) and estimated fabric area weight (kg m^{-2}). We also annotate the ground-truth fabric weight with a dashed line. As the plot indicates, the refinement converges towards the true area weight.

5.6 Conclusion

In this chapter, we have presented a method for measuring intrinsic and extrinsic physical parameters for cloth in the wind without ever observing real cloth before. The iterative measurement gradually improves by assessing the similarity between the current cloth simulation and the real-world observation. By leveraging only cloth simulations, we have described a method to train a physical similarity function. This enables measuring the physical correspondence between real and simulated data. To describe cloth dynamics, we have introduced a spectral decomposition layer that extracts the relevant features from the signal and generalizes from simulation to real observations. We compare the proposed method to prior work that considers flags in the wind and hanging cloth and obtain favorable results. For future work, given an appropriate physical embedding function, the proposed approach could be considered for other physical phenomena such as fire, smoke, fluid, foam or mechanical problems.

CHAPTER 6



LEARNING PHYSICAL PROPERTIES AND RELATIONSHIPS

6.1 Introduction

After exploring the value of recurrent motion for counting and physical property estimation, we shift gears to a more abstract notion of visual recurrence. In this chapter, we will use the reoccurring behavior of physical objects to infer physical properties and learn relationships.

In many cases, the best condensation of experience is to formulate laws. Physical laws formulating the basic principle of movement, social laws predicting human interaction, or physiological laws that tell us what motion of the limbs is permitted by the body. As humans, we learn these patterns implicitly by observing how things move under physical forces, how people interact in social scenes, and how the body moves. In general, computer vision algorithms focus on the appearance, for example by acquiring a variety of examples and learning in auto-encoder fashion what a natural interaction would be.

In this chapter, our goal is to take this one step further. We want to observe interactions and constrain regularities from natural movement in simple physical scenes. Where recent works in deep learning (Sanchez-Gonzalez et al., 2020; Y. Li et al., 2019; Mrowca et al., 2018) have focused on learning differentiable physics simulators for future state prediction, our emphasis is on understanding the physical knowledge acquired

This chapter is submitted for possible publication.

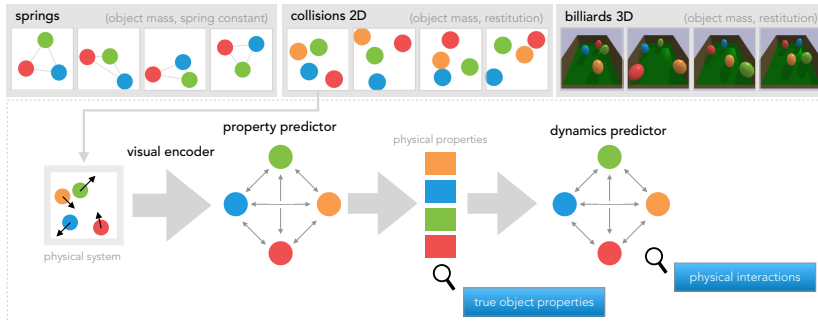


Figure 6.1. Given visual observations of some object-centric physical phenomena (top row), we consider the problem of inferring relevant intrinsic object properties and learning an implicit physical model. Our solution uses three networks that operate together: a *visual encoder* network, a *property predictor* and a *dynamics predictor*. After training, our model learns interpretable physical properties (e.g. mass and coefficient of restitution) and pairwise relationships (e.g. spring constant, damping and collisions).

by such models. Compared to existing work, we take an essential step: we start from the visual observation of the phenomenon and use visual regularities, rather than relying on object trajectories as specified directly in terms of position and velocities.

In specific applications, progress has been made in physical models of cloth (Liang et al., 2019; Bouman et al., 2013; Runia et al., 2020a), fluids (Ummenhofer et al., 2020; Y. Li et al., 2019) or collision systems (Sanchez-Gonzalez et al., 2020; Yi et al., 2020; Ehrhardt et al., 2018). However impressive the results, true generalization between different physical models is yet to come. The papers most akin to our effort learn implicit physical models using graph networks from object trajectories specified as a sequence of per-object positions and velocities (Zheng et al., 2018; Battaglia and Hamrick, 2018; Cranmer et al., 2019). We aim to learn such physical interactions directly from visual observations. In particular, we consider the task of extracting object-specific physical properties and pairwise interactions from three visually rendered physics simulations consisting of spring and collision systems (Figure 6.1; top row).

The primary goal is to learn an implicit physical model and intrinsic object properties directly from a sequence of visual observations. We do so through future state prediction in a latent object space that requires encoding intrinsic object properties as well as pairwise object interactions. One challenge is in the number of observational steps needed to

infer essential object properties such as the object’s mass. Furthermore, the unavailability of underlying per-object states as position-and-velocity sequence, prohibits formulating a regression loss between predicted and target states to optimize by learning.

Our solution uses graph networks (Scarselli et al., 2008; Battaglia and Hamrick, 2018; Battaglia et al., 2016; Gilmer et al., 2017; Chang et al., 2017) with contrastive learning (Kipf et al., 2020; LeCun et al., 2006; Runia et al., 2020a) to train the models in the absence of true object states as a supervision signal. Our model contains three parts that are jointly trained (Figure 6.1; bottom row): a *visual encoder* to encode video frames, a *property predictor* to infer intrinsic object properties from a sequence of latent factorized object states and a *dynamics predictor* to predict the system’s abstract future state based on the current state and object properties. This multi-stage approach builds upon the perception-prediction network (Zheng et al., 2018). However, we consider visual observations directly rather than the condensed trajectory of states. Condensation implies purpose and hence we aim to start on a more general route: the one starting from the visual observation. Consequently, we arrive at contrastive learning to train the models without direct state supervision. To demonstrate the interpretability of the learned object property vectors, we perform a correlation analysis with true object properties. Furthermore, we investigate whether the graph networks can learn pairwise physical interactions by a qualitative model dissection.

6.2 Related Work

Learning physical properties and relationships from observations is an active area of research (Battaglia et al., 2016; Kulkarni et al., 2015; Chang et al., 2017; Watters et al., 2017; Kipf et al., 2018; Mrowca et al., 2018; Zheng et al., 2018; Y. Li et al., 2019; Van Steenkiste et al., 2018; Ye et al., 2018; Jaques et al., 2020; Yi et al., 2020; Rempe et al., 2019; Sanchez-Gonzalez et al., 2020). In particular, *graph networks* (Gilmer et al., 2017; Battaglia and Hamrick, 2018) and its *interaction network* variant (Battaglia et al., 2016; Battaglia and Hamrick, 2018) have emerged as an highly-effective approach for modeling complex dynamics of object-oriented physical systems. Accurate modeling of particle systems opens the door to neural network-based differentiable physics simulators (Sanchez-Gonzalez et al., 2020; Mrowca et al., 2018;

Y. Li et al., 2019) that can run significantly faster than traditional physics simulators.

Most existing work considers the task of learning such a differentiable simulator from object trajectories, *e.g.* a sequence of positions, velocities and accelerations (Mrowca et al., 2018; Y. Li et al., 2019; Sanchez-Gonzalez et al., 2020). This assumes that the true internal state of all objects is known. In this chapter, we consider a scenario in which we only have access to visual observations. Several papers consider a similar problem (Watters et al., 2017; Van Steenkiste et al., 2018; J. Wu et al., 2017; Jaques et al., 2020; Ehrhardt et al., 2018; J. Wu et al., 2016). In particular, visual interaction networks (Watters et al., 2017) build an object-factorized visual representation of a short video clip and predict the next position and velocity of the objects. While the model considers visual inputs, the network is still trained with true states as direct supervision, limiting the applicability to scenarios in which such information is available. In contrast, we exclusively consider visual observations as both the network inputs and supervision signal. As existing simulation datasets (Fragkiadaki et al., 2016; Lerer et al., 2016; Ye et al., 2018; J. Wu et al., 2016; Groth et al., 2018; Ehrhardt et al., 2018; Yi et al., 2020) do not meet our demands of being multi-object environments with instance-specific properties and pairwise interactions, we generate three new physical simulation datasets.

Our work builds upon the graph network-based perception-prediction model (PPN) from Zheng et al., 2018. Their experiments demonstrate convincingly that graph-based neural architectures are able to infer physical properties from a sequence of position-velocity trajectories. We propose a generalization of the PPN which allows learning properties and dynamics exclusively from visual observations only. To enable this, we draw inspiration from the recent work of Kipf et al., 2020 and Chapter 5 both of which proposing contrastive learning to learn physical relationships without direct state supervision. Different from existing literature, our emphasis is on distilling object properties and physical relationships from the trained models, rather than focusing on training a differentiable physics simulator for unrolling physically accurate simulations using neural networks. This angle shares similarity with the analysis of Cranmer et al., 2019 that indicates that explicit physical relationships can be learned from graph networks.

6.3 Method

In this chapter, we consider the task of learning object-centric static *physical properties* and *relations* from a sequence of video frames $I_t \in \mathbb{R}^{C \times H \times W}$ displaying a particular physical phenomenon. In contrast to existing literature (Jaques et al., 2020; Watters et al., 2017; Sanchez-Gonzalez et al., 2020), our primary interest is not in training an accurate differentiable physics simulator. Rather, we emphasize on distilling information from the learned graph-based models.

The physical scenarios (Figure 6.1; top row) we consider are approximately Markovian, meaning that the system’s next state is fully-determined by its previous state. Therefore we can formulate our problem as future state prediction (Jaques et al., 2020; Zheng et al., 2018). An accurate prediction of the system’s future state will necessitate both physical property and relationship understanding and is, therefore, a suitable objective for the task. Our proposed solution builds upon the perception-prediction framework (Zheng et al., 2018). Crucially, we consider visual inputs and adopt contrastive learning as an unsupervised training signal in an abstract object state space. We start with a brief discussion of graph network preliminaries (Section 6.3.1). Next, we discuss our method’s individual network components (Section 6.3.2) followed by the use of contrastive learning (Section 6.3.3). An overview of our method is presented in Figure 6.2.

6.3.1 Graph Network Preliminaries

We consider object-oriented physical systems that can be embedded as a directed graph of interacting particles. Specifically, we consider graphs $G = (S, E)$ defined as a set of object features $S = \{\mathbf{s}^k\}_{k=1:K}$ with $\mathbf{s}^k \in \mathbb{R}^{D_s}$. And a set of edge features $E = \{\mathbf{e}^l, \rho^l, \sigma^l\}_{l=1:L}$ with $\mathbf{e}^l \in \mathbb{R}^{D_e}$ between the sender σ^l and receiver ρ^l node of the l -th edge. To learn pairwise physical relationships between interacting objects, we use *graph networks* (GN) as described in Scarselli et al., 2008; Gilmer et al., 2017; Battaglia and Hamrick, 2018.

The GN predicts the system’s next object states in a two-step process. First, an edge-oriented multi-layer perception MLP_{edge} computes pairwise interactions or *messages* $\mathbf{e}^{l'}$ between connected nodes. Next, for each object \mathbf{s}^k , the incoming interactions are summed to obtain aggregated messages $\mathbf{e}^{k'}$. Finally, a node-oriented function MLP_{node} predicts the

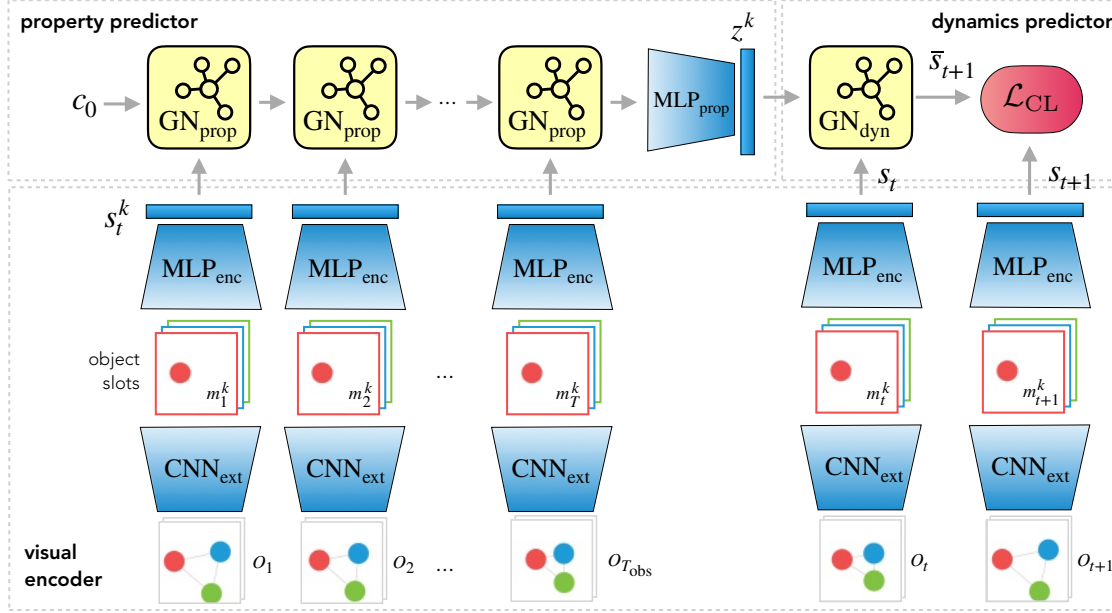


Figure 6.2. Overview of the proposed method. The model consists of three parts that are jointly trained using a contrastive loss: (1) a *visual encoder* to extract individual objects and embed them in some abstract state space; (2) a *property predictor* that learns physical properties from a sequence of observations; (3) a *dynamics predictor* that predicts the system's next abstract states. After learning, we perform correlation analysis to determine the relationship between true object attributes and learned properties.

next object features $\mathbf{s}^{k'}$ from the current object features and incoming aggregated messages. Our specific GN variant is an interaction network (Battaglia et al., 2016; Chang et al., 2017), taking the form of:

$$\mathbf{e}^{l'} = \text{MLP}_{\text{edge}}(\mathbf{e}^l, \mathbf{s}^{\rho^k}, \mathbf{s}^{\sigma^k}), \quad \mathbb{R}^{D_e} \times \mathbb{R}^{D_s} \times \mathbb{R}^{D_s} \rightarrow \mathbb{R}^{D'_e} \quad (6.1)$$

$$\mathbf{s}^{k'} = \text{MLP}_{\text{node}}(\bar{\mathbf{e}}^{k'}, \mathbf{v}^k), \quad \mathbb{R}^{D'_e} \times \mathbb{R}^{D_v} \rightarrow \mathbb{R}^{D'_v} \quad (6.2)$$

$$\bar{\mathbf{e}}^{k'} = \sum_{\{l: \rho^l=k\}} \mathbf{e}^{l'}, \quad (6.3)$$

where \mathbf{s}^{σ^k} and \mathbf{s}^{ρ^k} denote the object features of the sender and receiver respectively. This comprises one round of *message passing* denoted as high-level operation $\text{GN}(\cdot)$, which enables learning pairwise relations.

6.3.2 Model Architecture

Visual Encoder

Starting from a sequence of video frames, our model first decomposes the observations into a K *object slots*, where each slot $k = 1, \dots, K$ will correspond to a single object. From the object slots, we compute an object-oriented factorized state space $S = S_1 \times \dots \times S_K$ that will serve as input to both the property prediction networks and dynamics predictor. Within an observation episode, we assume that the objects are uniquely identifiable and that the number of objects remains the same. To present the network with implicit velocity information, we form an input observation O_t by stacking two consecutive frames over the channel dimension: $O_t = [I_t, I_{t+1}]$. We first use a convolutional network to predict K slots each corresponding to a single object: $m_t^k = [\text{CNN}_{\text{ext}}(O_t)]_k$ where subscript k denotes the selection of the k -th feature map. After extracting the object slots, we feed the feature maps to a feed-forward MLP to embed the objects into an abstract state space: $\mathbf{s}_t^k = \text{MLP}_{\text{enc}}(m_t^k)$. These abstract states will correspond to the object features in the graph network (6.1)-(6.2).

Property Predictor

The task of the property predictor is to infer the intrinsic object-specific properties for all objects conditioned on a sequence of factorized abstract states \mathbf{s}_t for $t = 1, \dots, T_{\text{obs}}$. For example, from an episode of colliding

balls, the property predictor should learn to encode the objects' mass and coefficients of restitution. To achieve this, we use a graph network $\text{GN}_{\text{prop}}(\cdot)$ recurrently for T_{obs} timesteps to learn from its physical interactions. Like the work by [Zheng et al., 2018](#), the recurrent architecture maintains an internal memory \mathbf{c}_t initialized as $\mathbf{c}_0 = \mathbf{0}$. At each step, the GN takes as input the memory vector \mathbf{c}_{t-1} concatenated with the current abstract state \mathbf{s}_t . After processing all observations, the static object properties will be estimated by conditioning another MLP on the final code vectors: $\bar{\mathbf{z}}^k = \text{MLP}_{\text{prop}}(\mathbf{c}_{T_{\text{obs}}}^k)$ with $\bar{\mathbf{z}}^k \in \mathbb{R}^{D_P}$. Finally, following [Zheng et al., 2018](#), we compute relative object properties \mathbf{z}^k by using the first object as a reference object. All together, the property prediction network takes the form of:

$$\mathbf{c}_t = \text{GN}_{\text{prop}}(\mathbf{c}_{t-1} \parallel \mathbf{s}_t) \quad \text{for } t = 1, \dots, T_{\text{obs}} \quad (6.4)$$

$$\bar{\mathbf{z}}^k = \text{MLP}_{\text{prop}}(\mathbf{c}_{T_{\text{obs}}}^k) \quad (6.5)$$

$$\mathbf{z}^k = \bar{\mathbf{z}}^k - \bar{\mathbf{z}}^1. \quad (6.6)$$

The concatenation \parallel is applied over the feature dimension and (6.6) expresses computing relative object properties by subtracting the reference object ($k = 1$); this ensures $\mathbf{z}^1 = \mathbf{0}$.

Dynamics Predictor

After inferring the object properties from T_{obs} visual input frames, the task of the dynamics predictor is to learn a graph-based physical model to predict the system's future state. The dynamics predictor considers an observation $O_{T_{\text{obs}}+1}$ and also starts by computing the factorized state representation \mathbf{s}_t^k using the visual encoder. Then, another graph network $\text{GN}_{\text{dyn}}(\cdot)$ predicts the next abstract factorized states by conditioning on the previous states and the property vectors \mathbf{z} :

$$\hat{\mathbf{s}}_{t+1}^k = \text{GN}_{\text{dyn}}(\mathbf{s}_t \parallel \mathbf{z}). \quad (6.7)$$

Consequently, the dynamics predictor has to learn per-object state transitions which require both static object properties and an implicit model of pairwise physical interactions. Jointly, the property and dynamics prediction network enable (1) estimating object-specific intrinsic properties; and (2) learning physical dynamics from visual observations only. While we could train the model for a single timestep prediction in the Marko-

vian setting, we found it helpful to train for multiple T_{pred} prediction timesteps to ensure the occurrence of potentially sparse collision events. What remains is how to optimize the networks as we only have access to abstract object states and visual observations.

6.3.3 Contrastive Learning

With no access to the per-object positions and velocities, we seek an alternative to the standard regression loss for predicting future object states. Existing work (J. Wu et al., 2016; Ye et al., 2018; Jaques et al., 2020) has proposed to decode predicted future states to pixel space by jointly training a decoder network. Based on our preliminary experiments, we found two problems with adding a decoder: (1) inferring object properties by observing video may require a substantial number of input frames, making it hard to sequentially decode all predictions due to memory constraints; (2) pixel-wise reconstructions overemphasize on low-level visual details rather than important physical dynamics. In light of this, we are inspired by recent works (Kipf et al., 2020; Runia et al., 2020a) and adopt *contrastive learning* (Hadsell et al., 2006) for training our model.

We use an energy-based contrastive loss (LeCun et al., 2006) in the abstract state space by comparing the predicted next state $\hat{\mathbf{s}}_{t+1}^k$ from the dynamics predictor (6.7), the “true” next state \mathbf{s}_{t+1}^k obtained from the visual encoder, and a randomly sampled negative state $\tilde{\mathbf{s}}_t^k$. The energy of a state-state tuple can be defined as $H = d(\text{GN}_{\text{dyn}}(\mathbf{s}_t \parallel \mathbf{z}), \mathbf{s}_{t+1}^k)$ where $d(\cdot, \cdot)$ denotes the squared Euclidean distance in embedding space \mathbb{R}^{D_s} . Following Kipf et al., 2020, we employ an energy-based hinge loss individually over the K object slots. Additionally, we also consider multiple prediction timesteps. This defines the positive energy H and negative energy \tilde{H} :

$$H = \frac{1}{K} \sum_{k=1}^K \sum_{t=1}^{T_{\text{pred}}} d(\text{GN}_{\text{dyn}}(\mathbf{s}_t \parallel \mathbf{z}), \mathbf{s}_{t+1}^k), \quad (6.8)$$

$$\tilde{H} = \frac{1}{K} \sum_{k=1}^K \sum_{t=1}^{T_{\text{pred}}} d(\tilde{\mathbf{s}}_t^k, \mathbf{s}_{t+1}^k). \quad (6.9)$$

The energy-based hinge loss then takes the form (LeCun et al., 2006):

$$\mathcal{L} = H + \max\left(0, \gamma - \tilde{H}\right), \quad (6.10)$$

which we compute as expectation over a mini-batch of examples. The margin γ is a hyper-parameter which we set $\gamma = 1$ throughout this chapter. At each step of the prediction, we sample negative states $\tilde{\mathbf{s}}_t^k$ by permuting the examples within the current mini-batch.

6.4 Experimental Setup

6.4.1 Physics Environments

There exist several physics simulations datasets (Fragkiadaki et al., 2016; Lerer et al., 2016; Ye et al., 2018; Yi et al., 2020; Zheng et al., 2018) but none of them meets all our criteria, most importantly being multi-object environments in which objects with varying physical properties participate in pairwise physical interactions. Therefore, we build three new physics simulation datasets (see Figure 6.1 for examples):

- **Springs:** The 2D springs simulation dataset features three balls, interconnected by springs of varying stiffness and damping factors. The continuous nature of the coupled spring relations enables efficient learning of physical interactions. The balls have a randomly chosen mass from $\{2, 5, 10\}$. The springs' stiffness values and damping coefficients are sampled from $\{20, 250, 500\}$ and $\{1, 2, 4\}$ respectively. The initial velocity magnitude is uniformly sampled from $[1000, 1500]$ and the springs' rest lengths are half the image size.
- **Collisions:** In this 2D dataset, four balls bounce off the walls and collide with other balls. This dataset is more difficult than the springs environment as the non-continuous nature of collisions makes it harder to infer object properties. The balls' masses and restitution coefficients are chosen from $\{2, 4.6, 7.3, 10\}$ and $\{.25, .5, .75, 1.0\}$ respectively. Each ball is given an initial velocity, with its magnitude uniformly sampled from $[1000, 1500]$.
- **Billiards 3D:** This 3D colliding billiard balls environment poses a greater difficulty due to its higher visual complexity in terms of appearance, shadows and perspective. The generation procedure is similar to the 2D collision dataset but we randomly sample the

material densities from $\{.2, .47, .73, 1\}$ and restitution coefficients from $\{.01, .34, .67, 1\}$.

In all environments, the object colors are randomly assigned such that they do not reveal information about the physical properties. For both collision datasets, we generate three subsets: one in which the mass varies, one in which the restitution varies and one in which both properties vary. Also, we ensure the possibility of inferring properties by rejecting episodes in which not all balls participate in at least one collision during the first T_{obs} timesteps. We generate the 2D datasets using PyMunk while we use PyBullet for the 3D billiards dataset. The 2D datasets are rendered at 64×64 with primitive drawing functions. The 3D dataset is rendered at 80×80 with PyBullet’s internal renderer. In all settings we use $T_{\text{obs}} = 30$ observation steps and $T_{\text{pred}} = 10$ prediction steps sampled at 20 fps. We generate 100k training episodes and 5k test episodes with different random seeds.

6.4.2 Model and Training Details

We use PyTorch (Paszke et al., 2019) for implementing all models. The slots extractor CNN_{ext} is implemented as a convolutional network with four Conv-BatchNorm-ReLU-MaxPool blocks and a final sigmoid activation. The resulting features maps serve as input for the object encoder MLP_{enc} , taking the form of a 3-layer MLP with 128 hidden units and ReLU activations. Both graph networks GN_{prop} and GN_{dyn} use the same configuration: the relation encoders are 4-layer MLPs with 128 hidden units, ReLU activations; the node encoders are similar but use only 3 layers. The final property encoder MLP_{prop} is a small 2-layer MLP with 32 hidden units. The abstract states have dimensionality $D_s = 4$, the edge features use $D_e = 4$ and the properties are $D_p = 16$ dimensional. The number of object slots K is manually set to match the physical environments. We optimize the networks with Adam (Kingma and Ba, 2015) until the training loss converges, which is typically in ± 300 epochs. The code and datasets will be made available.

6.4.3 Evaluation Metrics

Training with a contrastive loss prohibits direct evaluation of either position-velocity predictions or deviation in pixel space. During training, we therefore use the ranking-based assessment as suggested by Kipf et

al., 2020: the Hit@k metric is 1 if the predicted next state is within the k -nearest neighbors of the encoded true object state; and 0 otherwise. We compute the average over the validation dataset. We also report the mean reciprocal rank.

6.5 Experiments

6.5.1 Correlation Analysis of Properties from Vision

In the first experiment, we assess whether our model learns interpretable physical object properties from visual observations only. We expect the property predictor to learn meaningful object properties as they are required by the dynamics predictor for accurate future state prediction. We investigate the interpretability of learned physical properties by correlation analysis (Bouman et al., 2013; Zheng et al., 2018) between the property vectors and true object simulation properties. First, we extract the property vectors $\mathbf{z}^k \in \mathbb{R}^{D_p}$ for the entire test set and perform principal component analysis to project the property vectors into a lower-dimensional space. Next, we train a linear regression model between the first principal components (PC_{1,2}) of the property vectors and the true object properties for the particular physical environment. To quantify the interpretability of the learned property vectors, we report the coefficient of determination (R^2) between true object properties and the principal components of the property vectors.

As a strong baseline, we also train the original PPN model from Zheng et al., 2018 with a supervised L_2 regression loss using the objects' true position and velocity. We consider this baseline as an upper bound for our model as it uses the exact object position-velocity information as a direct supervision signal. A consequence of using our slots encoder with a contrastive loss is that the learned object slots will be in random (but fixed) order (see Figure 6.3). Therefore, during the correlation analysis, we check all object permutations and report the highest R^2 value. We always observe that one of the permutations is the “correct” one yielding a significantly higher score than the others, indicating a consistent order assignment of object slots.

We report the results in Figure 6.4 and make several observations. For our model on the springs environment, we find that the first principal component strongly correlates with the log of true mass. The continu-

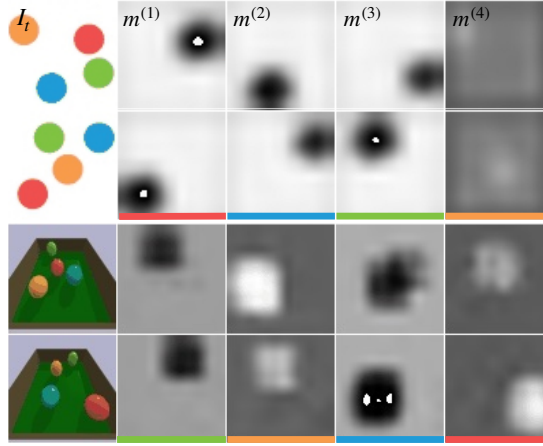


Figure 6.3. Visualization of the learned object slots for two examples from both collision scenes. The encoder learns a random order of object colors which remain fixed during training.

ous nature of the spring interaction enables the property predictor to effectively learn the relative object masses. In line with expectation, we also find a zero correlation with other irrelevant object properties such as the ball radius, friction and color. For the 2D collision dataset, we find slightly lower correlation coefficients, suggesting that the sparsity of collision interactions poses a greater difficulty. Specifically, for the subset with restitution coefficient-variations only, we observe that the balls lose their kinetic energy more quickly due to the inelastic collisions. Consequently, as the balls’ average velocity decreases more rapidly, the total number of collisions is lower and it becomes more difficult to estimate the material properties. During training, we periodically compute the R^2 coefficients and find that on the collision datasets, our contrastive models only start to learn the object properties after a considerable amount of epochs. Only for prediction timesteps with a collision, the dynamics predictor requires an accurate estimate of the object properties. This suggests that some form of temporal attention to emphasize on collisions could improve training efficiency. For the Billiards 3D dataset, we observe a significant gap between our visual model and the upper bound as set by the state-supervised model. Although the visual encoder quickly learns to decompose the 3D scene into its distinct objects slots (Figure 6.3), estimating the physical properties from the sequence of latent states poses a greater difficulty than for its 2D counterpart.

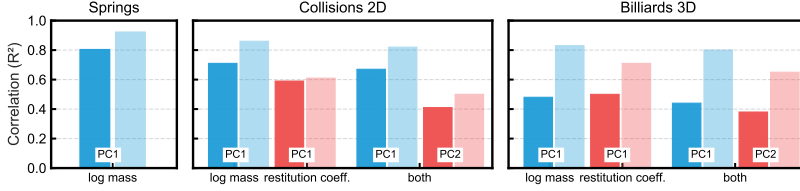


Figure 6.4. Correlation analysis (R^2) between learned principal components (PC_{1,2}) of property vectors and true object properties. ■ / ■ denotes correlation with true mass for our visual method (dark hue) and non-visual PPN (light hue) respectively. ■ / ■ similarly for correlation with restitution coefficients. For the springs, we find that our network learns an interpretable representation of the object’s mass.

6.5.2 Learning Physical Relationships from Vision

Next, we perform a qualitative analysis of the relationships learned by the dynamics predictor. To extract physical relationships between the objects, we draw inspiration from [Cranmer et al., 2019](#) which demonstrates that graph networks can learn the force law from n -body object trajectories. Our aim is to find evidence that the graph networks can learn Hooke’s law or an implicit collision model from visual observations only. Specifically, we consider the information concealed in the *aggregated messages* $\bar{\mathbf{e}}^{k'}$ ((6.3)) from the dynamics predictor. To investigate this, we plot the first principal components of the aggregated messages in the dynamics predictor for our model trained on the springs and collision environment. We train the particular models for $T_{\text{pred}} = 10$ steps and subsequently unroll the model for 40 timesteps and record the message vectors $\bar{\mathbf{e}}^{k'}$ at each step for all objects. Two example plots of the aggregated messages are displayed in [Figure 6.5](#). For the springs environment, we clearly observe a sinusoidal signal profile with damping, suggesting that the graph network has indeed learned an implicit formulation of Hooke’s law for oscillating springs. The interpretation of the messages for the collision model is less evident. Rather than learning impulses responses upon collisions, we observe a more continuous nature of the message vectors. Perhaps, the dynamics predictor internally learns a continuous representation to encode the distance between other balls and the container walls.

Confirming the observations made by [Cranmer et al., 2019](#), we also find that setting the edge-dimensionality D_e to minimally span the

world’s dimensionality ($D_e = 2$ for two-dimensional environments) improves the dynamics predictor in terms of the ranking metrics. Increasing the edge dimensionality could lead to the network learning complex interactions resulting in decreased generalization performance.

6.6 Conclusion

In this chapter, we have considered learning physical object properties and physical relationships from visual observations only. The contrastive loss in the abstract state space yields a viable alternative to the commonly used regression loss in the absence of true per-object state information such as positions and velocities. Experiments on our three physics environments indicate that our visual property predictor and dynamics predictor can discover interpretable physics properties such as mass, coefficients of restitution and spring constants. One of the main challenges is to learn object properties from sparsely occurring collision events. From a visual perspective, our method is currently unable to handle non-identifiable objects and moreover has no ability to deal with objects entering or leaving the environment. Future work in this direction could include learning non-physical properties and relations in unsupervised settings such as the movement of human limbs, sports tactics or pedestrian interactions.

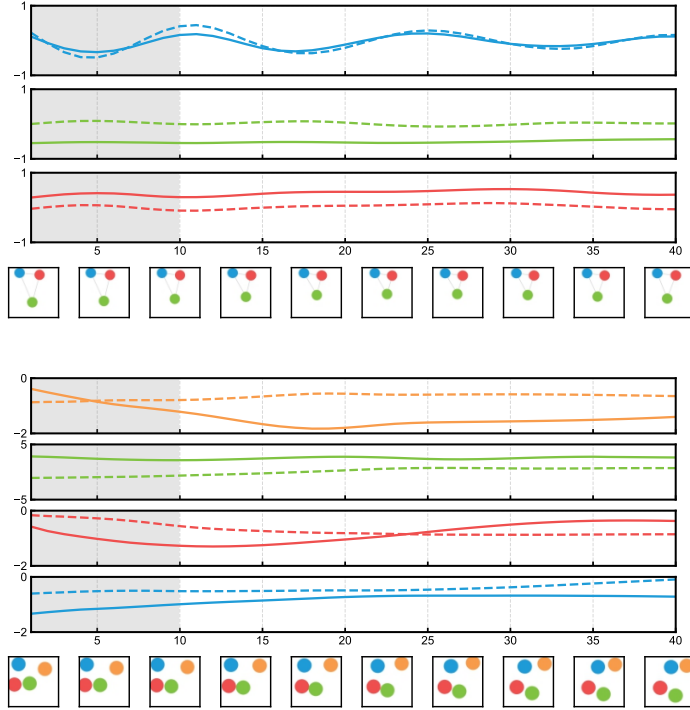


Figure 6.5. For a trained model on the springs (left) and collision environments (right), we plot the first two principal components of the aggregated messages $\bar{\mathbf{e}}^{k'}$ for each of the objects. The model was trained for 10 steps and we unroll the model for 40 timesteps. The first ten frames of the episode, corresponding to the shaded region, are displayed below the plot. For the springs, we observe a clear oscillating signal.

CHAPTER 7



CONCLUSION

In this thesis, we have explored recurrent motion in video from different perspectives. Our story was structured as two parts: Part I “*Recurrent Motion in Vision*” has proposed new theory for periodic motion in video, followed by two novel methods for counting and localization of repetitive motion patterns; Part II “*Recurrent Physical Dynamics*” has concentrated on the appearance and value of repetitive motion in physical phenomena. Having addressed the main body of work in these two parts, we here revisit the research questions from [Chapter 1](#) by answering them based on our findings. Furthermore, we address the current limitations of our research and suggest potential directions for future research.

7.1 Research Questions and Contributions

Here we return to the research questions in the two parts:

Part I. Recurrent Motion in Vision

Research Question 1: What periodic motion types exist for objects moving in 3D space and what are their appearance types on the 2D image plane?

In [Chapter 2](#) ([Runia et al., 2019](#)), we have attempted to answer this question by exploring the origins and grouping of periodic motion as tied to an object moving in 3D space. We have followed a differential approach, starting from the divergence, gradient and curl components of the 3D flow field. From the decomposition of the motion field and its temporal dynamics, we have derived three motion types and three motion continuities to arrive at 3×3 fundamental cases of intrinsic periodicity in 3D.

For the 2D perception of 3D intrinsic periodicity, the observer's viewpoint can be somewhere in the continuous range between two viewpoint extremes. Ultimately, from the flow field decomposition and the two distinct viewpoint cases, we have arrived at 18 fundamental cases for the 2D perception of 3D intrinsic periodic motion. We have also introduced the concepts of non-stationary and non-static periodic motion. Under realistic circumstances, repetitive motion is rarely perfectly periodic in its cycle length and its appearance may change due to camera motion or a gradual change in the motion pattern.

While [Chapter 2](#) provides a step in the exhaustive categorization of periodic motion, interesting open questions remain. For example, we have not considered the superposition of multiple motion patterns such as spiraling motion. Neither have we discussed the possible, but unlikely, scenario in which different orientations in 3D space are counterbalanced, such as contraction in one dimension and expansion in another. The shear component of the flow field was omitted by the observation that it rarely occurs under standard conditions on a macroscopic scale. Nonetheless, an exhaustive study of shear in the motion field is relevant for microscopic scenarios or industrial applications in which large tension is involved (*e.g.* cutting sheets of metal). Lastly, we have not dedicated special interest to dynamic textures which may induce recurrent motion. Future work could attempt to unify our motion-based theory with a study on dynamic texture recognition ([Hadji and Wildes, 2017](#)) or Eulerian motion analysis ([H.-Y. Wu et al., 2012](#)).

Research Question 2: How can we detect, localize and count repetitive motion in arbitrary realistic video?

We answer this question in the affirmative as supported by the two repetition estimation methods presented in [Chapter 3](#) ([Runia et al., 2018](#)) and [Chapter 4](#) ([Runia et al., 2019](#)). Both methods have built upon the differential analysis of the motion field as stipulated by the theory. Unlike existing work, we have emphasized the appearance of non-stationary motion in video. Our solution uses the continuous wavelet transform for handling weakly-periodic motion, dealing with accelerations or detecting transient phenomena. The initial method presented in [Chapter 3](#) has relied on a decoupled state-of-the-art method for localizing the foreground motion. The aim of such methods is to accurately segment the

foreground motion, but it is unclear whether they are optimal for estimating repetition in video. We have investigated this in [Chapter 4](#) by the proposition of an improved method that directly uses the wavelet power spectrum for spatially localizing repetitive motion. We conclude that this specialized localization indeed outperforms decoupled foreground motion localization method on the task of repetition counting. Furthermore, the improved method of [Chapter 4](#) can handle dynamic viewpoint changes or gradual changes in the motion by combining all differential flow representations rather than selecting the one with the strongest response as was our solution in [Chapter 3](#). To evaluate our methods in comparison to existing work, we have constructed a novel video dataset for repetition counting. Compared to existing datasets, our videos emphasize real-world circumstances with increased motion diversity, non-stationarity and camera motion.

While our method requires no learning, it can outperform an existing deep learning method ([Levy and Wolf, 2015](#)) for repetition counting. We believe that hybrid methods, which combine both parallel efforts, could be interesting for future research. In particular, more robust motion representations could be learned from the data directly given that a sufficient amount of real-world training data is available. Nonetheless, we believe that a strong inductive bias as delivered by the theory can reduce the need for training data and may perform equally well in most situations. Our contributions in this chapter have not explored the correlation between particular frequency responses and the specific action classes. Existing work ([Johansson, 1973](#); [Cheng et al., 2004](#)) indicates that recurrent motion in video could serve as a strong feature descriptor for human action classification. In light of our novel methods for localizing and estimating repetition in video, we believe it is worthwhile to reconsider this question in future work. On the task of counting, we have identified the common failure case of over-counting the number of repetitions with a factor two when the movement of both limbs produces a similar motion pattern (for example swimming front-crawl or rowing). One line of future work is solving this problem by adding heuristics or predicting multiple counts by further analysis of the wavelet scalogram.

Part II. Recurrent Physical Dynamics

Research Question 3: Can we use recurrent motion in video for real-world physical measurements?

Our insights and ability to localize, detect and count repetitions in video opens the door to increasingly complex applications of recurrent motion in video. In [Chapter 5 \(Runia et al., 2020a\)](#) we have identified the oscillating motion of cloth in the wind as a case study of recurring physical dynamics. Specifically, we have proposed to measure external forces (*i.e.* wind speed) and intrinsic properties (*i.e.* fabric density) from a video of flapping cloth without any prior real-world observations. Our solution uses contrastive learning to train a Siamese network that quantifies the physical similarity between two video clips. During training, we leverage cloth physics simulations and 3D rendering to construct positive and negative training pairs to learn a structured physical similarity embedding space. Inspired by the previous chapters, the network architecture uses a new spectral decomposition layer that models the spatial distribution of spectral energy over the surface of the cloth. In contrast to using raw pixel information, this results in a more robust domain transfer to real-world data. During inference, the measurement takes the form of an iterative refinement procedure in which we gradually adjust the physics simulations to increase physical similarity with the true observation. Ultimately, this produces a measurement of simulation parameters from the real-world and its corresponding cloth simulation. The empirical evaluation on our novel dataset of flag measurements and the existing hanging cloth dataset ([Bouman et al., 2013](#)) indicate that our approach is able to measure real-world physical parameters from video without having seen real cloth before.

An important drawback of our current solution is the high computational cost involved with running the iterative measurements due to the gradual refinement of the simulations. Accurate cloth simulations are expensive due to the fabric’s high level of detail and the intricate cloth-force coupling ([Liang et al., 2019](#)). Consequently, our measurement process cannot run in real-time nor is it suitable for mobile devices. Future work could focus on improving computational efficiency by estimating the initial simulation parameters and constraining the parameter search space. The empirical evaluation of our spectral decomposition

layer only considers the narrow domain of cloth; future research could consider the new neural network layer for repetition estimation, human action classification or dynamic texture recognition. From a methodological perspective, the proposed algorithm for iteratively measuring physical parameters from video without prior real-world observations could be considered for other physical phenomena such as water, smoke or mechanical problems.

Research Question 4: What do recurrent object interactions reveal about physical object properties and dynamics?

In [Chapter 6 \(Runia et al., 2020b\)](#) we have proposed to learn intrinsic object properties and relationships from the repeated observation of particular object-centric physical interactions. Existing approaches have either estimated the parameters of known physical models or have ignored the problem’s visual aspect. Our solution considers learning intrinsic object properties and their pairwise physical relationships from visual observations only. This was challenging as (1) we had to factorize the visual imagery into individual objects; and (2) inferring object properties may demand to observe the phenomenon for a longer period. The suggested solution comprises three models that operate together: the visual encoder disentangles the observations into an abstract factorized state representation. Based on a sequence of abstract object states, the property predictor, implemented as graph network, encodes physically relevant object properties. And, the dynamics predictor, another graph network, is conditioned on the object properties and the current observation to predict the system’s future state. With no access to true object states and to alleviate the need for a loss in pixel space, we have used contrastive learning for joint training of the networks on three physics environments. Post-training model dissection using correlation analysis indicates that our models can indeed learn physical object representations corresponding to their true values.

We found that learning physical properties from video is particularly difficult for physical environments with sparse interactions such as collisions. The number of timesteps that contain valuable information for estimating physical properties is limited. This translates in an excessively long training duration. In contrast, the continuous nature of spring systems dramatically reduces training time. Considering these

observations it could be worthwhile to explore the use of temporal attention for concentrating on object interactions. A different line of work has focused on building differentiable physics simulation engines using graph networks (e.g. [Sanchez-Gonzalez et al., 2020](#)). Improving the ability to convincingly simulate real-world physical environments demands to increase the number of particles. To accommodate this, recent works impose a hierarchical structure on the interaction graph ([Mrowca et al., 2018](#)) to reduce the number of pairwise relations for computational efficiency. In contrast, our work focuses on learning physical properties from *visual* observations. This requires decomposing the visual scene into individual objects which currently requires one or more objects slots (*i.e.* feature maps) per object. Future work could explore learning large-scale physical dynamics from visual observations by reducing computational complexity and memory demands as they currently exist in our method.

7.2 Closing Remarks

What it means to see includes the ability to recognize, localize and count repetitive motion in the visual world around us. The theme of recurrent motion has taken us on a journey through a variety of topics in computer vision. This thesis has first and foremost exposed the pervasiveness of recurrent motion in computer vision. Recurrent motion appears in a broad range of human actions, in natural scenes, mechanical systems and in physical phenomena. Despite its ubiquity, no large video datasets of recurrent motion exist. This has given us the opportunity to construct new datasets and learn from the assembling process. Furthermore, the lack of large training datasets has forced us to deliver solutions that remain meaningful in the absence of data. Throughout the journey, this has strengthened our belief that incorporating prior knowledge into computer vision algorithms by means of inductive biases or specialized architectural novelties can improve the solutions, reduce the dependence on training data and increase generalization to unseen circumstances.



BIBLIOGRAPHY

- Abraham, Ralph, Jerrold E Marsden, and Tudor Ratiu (1988). *Manifolds, Tensor Analysis, and Applications*. Springer.
- Adelson, Edward H (2001). “On seeing stuff: the perception of materials by humans and machines”. In: *Human Vision and Electronic Imaging*.
- Albu, A Branzan, Robert Bergevin, and Sébastien Quirion (2008). “Generic temporal segmentation of cyclic human motion”. In: *Pattern Recognition*.
- Azy, Ousman and Narendra Ahuja (2008). “Segmentation of periodically moving objects”. In: *International Conference on Pattern Recognition (ICPR)*.
- Baraff, David and Andrew Witkin (1998). “Large steps in cloth simulation”. In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.
- Battaglia, Peter, Razvan Pascanu, Matthew Lai, and Danilo Jimenez Rezende (2016). “Interaction networks for learning about objects, relations and physics”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Battaglia, Peter W and Jessica Hamrick (2018). “Relational inductive biases, deep learning, and graph networks”. In: *arXiv:1806.01261*.
- Belongie, Serge and Josh Wills (2006). “Structure from Periodic Motion”. In: *Spatial Coherence for Visual Motion Analysis*. Springer.
- Bergstra, James and Yoshua Bengio (2012). “Random search for hyperparameter optimization”. In: *Journal of Machine Learning Research (JMLR)*.

- Bhat, Kiran S, Christopher D Twigg, Jessica K Hodgins, Pradeep K Khosla, Zoran Popović, and Steven M Seitz (2003). "Estimating cloth simulation parameters from video". In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.
- Bisley, James W (2011). "The neural basis of visual attention". In: *The Journal of physiology*.
- Blender (2018). *Blender - 3D Modelling and Rendering*. Blender Foundation.
- Bouman, Katherine L, Bei Xiao, Peter Battaglia, and William T Freeman (2013). "Estimating the material properties of fabric from video". In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.
- Brandon, Melissa and Jenny R Saffran (2011). "Apparent motion enhances visual rhythm discrimination in infancy". In: *Attention, Perception, & Psychophysics*.
- Briassouli, Alexia and Narendra Ahuja (2007). "Extraction and analysis of multiple periodic motions in video sequences". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.
- Bridson, Robert, Sebastian Marino, and Ronald Fedkiw (2005). "Simulation of clothing with folds and wrinkles". In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.
- Brock, Andrew, Jeff Donahue, and Karen Simonyan (2019). "Large scale gan training for high fidelity natural image synthesis". In: *International Conference on Learning Representations (ICLR)*.
- Bromley, Jane, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah (1994). "Signature verification using a siamese time delay neural network". In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Burghouts, Gertjan J and J-M Geusebroek (2006). "Quasi-periodic spatiotemporal filtering". In: *IEEE Transactions on Image Processing (TIP)*.
- Burt, Peter and Edward Adelson (1983). "The Laplacian pyramid as a compact image code". In: *IEEE Transactions on Communications*.

- Canny, John (1986). "A computational approach to edge detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.
- Cardona, Jennifer L, Michael F Howland, and John O Dabiri (2019). "Seeing the Wind: Visual Wind Speed Prediction with a Coupled Convolutional and Recurrent Neural Network". In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Chang, Michael B, Tomer Ullman, Antonio Torralba, and Joshua B Tenenbaum (2017). "A compositional object-based approach to learning physical dynamics". In: *International Conference on Learning Representations (ICLR)*.
- Chen, Ming-yu and Alexander Hauptmann (2009). *MoSIFT: Recognizing human actions in surveillance videos*. Tech. rep. CMU-CS-09-161. Carnegie Mellon University.
- Chen, Ting, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton (2020). "A simple framework for contrastive learning of visual representations". In: *arXiv preprint arXiv:2002.05709*.
- Cheng, Fangxiang, William Christmas, and Josef Kittler (2004). "Periodic human motion description for sports video databases". In: *International Conference on Pattern Recognition (ICPR)*.
- Chetverikov, Dmitry and Sándor Fazekas (2006). "On Motion Periodicity of Dynamic Textures". In: *British Machine Vision Conference (BMVC)*.
- Covell, Michele, Shumeet Baluja, and Michael Fink (2006). "Advertisement detection and replacement using acoustic and visual repetition". In: *IEEE Workshop on Multimedia Signal Processing*.
- Craik, Kenneth James Williams (1967). *The Nature of Explanation*. CUP Archive.
- Cranmer, Miles D, Rui Xu, Peter Battaglia, and Shirley Ho (2019). "Learning Symbolic Physics with Graph Networks". In: *NeurIPS Workshops*.
- Cutler, Ross and Larry S. Davis (2000). "Robust real-time periodic motion detection, analysis, and applications". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

- Davis, James, Aaron Bobick, and Whitman Richards (2000). "Categorical representation and recognition of oscillatory motion patterns". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Ding, Shengyong, Liang Lin, Guangrun Wang, and Hongyang Chao (2015). "Deep feature learning with relative distance comparison for person re-identification". In: *Pattern Recognition*.
- Dwibedi, Debidatta, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman (2020). "Counting Out Time: Class Agnostic Video Repetition Counting in the Wild". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Ehrhardt, Sebastien, Aron Monszpart, Niloy Mitra, and Andrea Vedaldi (2018). "Unsupervised intuitive physics from visual observations". In: *Asian Conference on Computer Vision (ACCV)*.
- Eloy, Christophe, Romain Lagrange, Claire Souilliez, and Lionel Schouveiler (2008). "Aeroelastic instability of cantilevered flexible plates in uniform flow". In: *Journal of Fluid Mechanics*.
- Feichtenhofer, Christoph, Haoqi Fan, Jitendra Malik, and Kaiming He (2019). "Slowfast networks for video recognition". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Fragkiadaki, Katerina, Pulkit Agrawal, Sergey Levine, and Jitendra Malik (2016). "Learning visual predictive models of physics for playing billiards". In: *International Conference on Learning Representations (ICLR)*.
- Fukushima, Kunihiro (1980). "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position". In: *Biological cybernetics*.
- Gilmer, Justin, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl (2017). "Neural message passing for quantum chemistry". In: *International Conference on Machine Learning (ICML)*.
- Goldenberg, Roman, Ron Kimmel, Ehud Rivlin, and Michael Rudzsky (2005). "Behavior classification by eigendecomposition of periodic motions". In: *Pattern Recognition*.

- Grossmann, Alexander and Jean Morlet (1984). "Decomposition of Hardy functions into square integrable wavelets of constant shape". In: *Journal on Mathematical Analysis*.
- Groth, Oliver, Fabian B Fuchs, Ingmar Posner, and Andrea Vedaldi (2018). "Shapestacks: Learning vision-based physical intuition for generalised object stacking". In: *European Conference on Computer Vision (ECCV)*.
- Haddon, John and David Forsyth (1998). "Shading primitives: Finding folds and shallow grooves". In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.
- Hadji, Isma and Richard P Wildes (2017). "A spatiotemporal oriented energy network for dynamic texture recognition". In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.
- Hadsell, Raia, Sumit Chopra, and Yann LeCun (2006). "Dimensionality reduction by learning an invariant mapping". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Deep residual learning for image recognition". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Hegarty, Mary (2004). "Mechanical reasoning by mental simulation". In: *Trends in Cognitive Sciences*.
- Henriques, João F, Rui Caseiro, Pedro Martins, and Jorge Batista (2012). "Exploiting the circulant structure of tracking-by-detection with kernels". In: *European Conference on Computer Vision (ECCV)*.
- Hermans, Alexander, Lucas Beyer, and Bastian Leibe (2017). "In defense of the triplet loss for person re-identification". In: *arXiv preprint arXiv:1703.07737*.
- Horn, Berthold KP and Brian G Schunck (1981). "Determining optical flow". In: *Techniques and Applications of Image Understanding*.
- Hove, Michael J, John R Iversen, Allen Zhang, and Bruno H Repp (2013). "Synchronization with competing visual and auditory rhythms: bouncing ball meets metronome". In: *Psychological Research*.

- Huang, Shiyao, Xianghua Ying, Jiangpeng Rong, Zeyu Shang, and Hongbin Zha (2016). "Camera calibration from periodic motion of a pedestrian". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Huang, Wei-Xi and Hyung Jin Sung (2010). "3D simulation of a flapping flag in a uniform flow". In: *Journal of Fluid Mechanics*.
- Ilg, Eddy, Nikolaus Mayer, T Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox (2017). "FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Jain, Mihir, Herve Jegou, and Patrick Bouthemy (2013). "Better exploiting motion for better action recognition". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- James, Stephen, Paul Wohlhart, Mrinal Kalakrishnan, Dmitry Kalashnikov, Alex Irpan, Julian Ibarz, Sergey Levine, Raia Hadsell, and Konstantinos Bousmalis (2019). "Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Jaques, Miguel, Michael Burke, and Timothy Hospedales (2020). "Physics-as-inverse-graphics: Joint unsupervised learning of objects and physics from video". In: *International Conference on Learning Representations (ICLR)*.
- Johansson, Gunnar (1973). "Visual perception of biological motion and a model for its analysis". In: *Perception & Psychophysics*.
- Karpathy, Andrej, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei (2014). "Large-scale video classification with convolutional neural networks". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Karvounas, Giorgos, Iason Oikonomidis, and Antonis Argyros (2019). "ReActNet: Temporal Localization of Repetitive Activities in Real-World Videos". In: *arXiv preprint arXiv:1910.06096*.

- Kingma, Diederik P and Jimmy Ba (2015). “Adam: A method for stochastic optimization”. In: *International Conference on Learning Representations (ICLR)*.
- Kipf, Thomas, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel (2018). “Neural Relational Inference for Interacting Systems”. In: *International Conference on Machine Learning (ICML)*.
- Kipf, Thomas, Elise van der Pol, and Max Welling (2020). “Contrastive Learning of Structured World Models”. In: *International Conference on Learning Representations (ICLR)*.
- Klaser, Alexander, Marcin Marszałek, and Cordelia Schmid (2008). “A spatio-temporal descriptor based on 3d-gradients”. In: *British Machine Vision Conference (BMVC)*.
- Koenderink, J and A van Doorn (1975). “Invariant Properties of the Motion Parallax Field due to the Movement of Rigid Bodies Relative to an Observer”. In: *International Journal of Optics*.
- Koenderink, Jan J (1984). “The structure of images”. In: *Biological cybernetics*.
- (1988). “Scale-time”. In: *Biological Cybernetics*.
- Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). “ImageNet classification with deep convolutional neural networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Kulkarni, Tejas D, William F Whitney, Pushmeet Kohli, and Josh Tenenbaum (2015). “Deep convolutional inverse graphics network”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Laptev, Ivan, Serge J Belongie, Patrick Perez, and Josh Wills (2005). “Periodic motion detection and segmentation via approximate sequence alignment”. In: *International Conference on Computer Vision (ICCV)*.
- LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton (2015). “Deep learning”. In: *Nature*.
- LeCun, Yann, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel (1989). “Back-propagation applied to handwritten zip code recognition”. In: *Neural Computation*.

- LeCun, Yann, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang (2006). "A tutorial on energy-based learning". In: *Predicting Structured Data*.
- Lerer, Adam, Sam Gross, and Rob Fergus (2016). "Learning physical intuition of block towers by example". In: *International Conference on Machine Learning (ICML)*.
- Levy, Ofir and Lior Wolf (2015). "Live Repetition Counting". In: *International Conference on Computer Vision (ICCV)*.
- Li, Hui, Yan Bao, Ernst Pöppel, and Yi-Huang Su (2014). "A unique visual rhythm does not pop out". In: *Cognitive processing*.
- Li, Wenbin, Seyedmajid Azimi, Aleš Leonardis, and Mario Fritz (2016). "To fall or not to fall: A visual approach to physical stability prediction". In: *arXiv preprint arXiv:1604.00066*.
- Li, Xiu, Hongdong Li, Hanbyul Joo, Yebin Liu, and Yaser Sheikh (2018). "Structure from Recurrent Motion: From Rigidity to Recurrency". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Li, Yunzhu, Jiajun Wu, Russ Tedrake, Joshua B Tenenbaum, and Antonio Torralba (2019). "Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids". In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Liang, Junbang, Ming Lin, and Vladlen Koltun (2019). "Differentiable Cloth Simulation for Inverse Problems". In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Lindeberg, Tony (2017). "Dense scale selection over space, time and space-time". In: *Journal on Imaging Sciences*.
- Liu, Fang and Rosalind W Picard (1998). "Finding periodicity in space and time". In: *International Conference on Computer Vision (ICCV)*.
- Lu, ChunMei and Nicola J Ferrier (2004). "Repetitive motion analysis: Segmentation and event classification". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.
- Lucas, Bruce D and Takeo Kanade (1981). "An iterative image registration technique with an application to stereo vision". In: *International Joint Conference on Artificial Intelligence (IJCAI)*.

- Mallat, Stephane G (1989). "A theory for multiresolution signal decomposition: the wavelet representation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.
- Marr, David (1982). "Vision: A computational investigation into the human representation and processing of visual information". In:
- Marr, David and Herbert Keith Nishihara (1978). "Representation and recognition of the spatial organization of three-dimensional shapes". In: *Proceedings of the Royal Society of London*.
- Mayer, Nikolaus, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox (2016). "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Meka, Abhimitra, Maxim Maximov, Michael Zollhoefer, Avishek Chatterjee, Hans-Peter Seidel, Christian Richardt, and Christian Theobalt (2018). "Lime: Live intrinsic material estimation". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Mottaghi, Roozbeh, Hessam Bagherinezhad, Mohammad Rastegari, and Ali Farhadi (2016a). "Newtonian scene understanding: Unfolding the dynamics of objects in static images". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Mottaghi, Roozbeh, Mohammad Rastegari, Abhinav Gupta, and Ali Farhadi (2016b). "'What happens if...': Learning to Predict the Effect of Forces in Images". In: *European Conference on Computer Vision (ECCV)*.
- Mrowca, Damian, Chengxu Zhuang, Elias Wang, Nick Haber, Li F Fei-Fei, Josh Tenenbaum, and Daniel L Yamins (2018). "Flexible neural representation for physics prediction". In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Narain, Rahul, Armin Samii, and James F O'Brien (2012). "Adaptive anisotropic remeshing for cloth simulation". In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.

- Oh, ChangYong, Efstratios Gavves, and Max Welling (2018). “BOCK: Bayesian Optimization with Cylindrical Kernels”. In: *International Conference on Machine Learning (ICML)*.
- Oppenheim, Alan V (1999). *Discrete-time signal processing*. Pearson Education.
- Otsu, Nobuyuki (1979). “A threshold selection method from gray-level histograms”. In: *IEEE Transactions on Systems, Man, and Cybernetics*.
- Papazoglou, Anestis and Vittorio Ferrari (2013). “Fast object segmentation in unconstrained video”. In: *International Conference on Computer Vision (ICCV)*.
- Papert, Seymour A (1966). “The summer vision project”. In:
- Paszke, Adam, Sam Gross, Francisco Massa, and Adam Lerer (2019). “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Peng, Xingchao, Ben Usman, Neela Kaushik, Dequan Wang, Judy Hoffman, and Kate Saenko (2018). “Visda: A synthetic-to-real benchmark for visual domain adaptation”. In: *CVPR Workshops*.
- Pogalin, Erik, Arnold W M Smeulders, and Andrew H C Thean (2008). “Visual quasi-periodicity”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Polana, Ramprasad and Randal C Nelson (1997). “Detection and recognition of periodic, nonrigid motion”. In: *International Journal of Computer Vision (IJCV)*.
- Provot, Xavier et al. (1995). “Deformation constraints in a mass-spring model to describe rigid cloth behaviour”. In: *Graphics Interface*.
- Ran, Yang, Isaac Weiss, Qinfen Zheng, and Larry S Davis (2007). “Pedestrian detection via periodic motion analysis”. In: *International Journal of Computer Vision (IJCV)*.
- Rempe, Davis, Srinath Sridhar, He Wang, and Leonidas Guibas (2019). “Learning Generalizable Final-State Dynamics of 3D Rigid Objects”. In: *CVPR Workshops*.

- Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun (2015). “Faster r-cnn: Towards real-time object detection with region proposal networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Repp, Bruno H and Amandine Penel (2004). “Rhythmic movement is attracted more strongly to auditory than to visual rhythms”. In: *Psychological research*.
- Revaud, Jerome, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid (2015). “EpicFlow: Edge-preserving interpolation of correspondences for optical flow”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Runia, Tom F H, Kirill Gavrilyuk, Cees G M Snoek, and Arnold W M Smeulders (2020a). “Cloth in the Wind: A Case Study of Estimating Physical Measurement through Simulation”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Runia, Tom F H, Cees G M Snoek, and Arnold W M Smeulders (2018). “Real-World Repetition Estimation by Div, Grad and Curl”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- (2020b). “Learning Physical Properties and Relationships from Visual Observations”. In: *Under Review*.
- Runia, Tom F H, Cees G M Snoek, and Arnold W M Smeulders (2019). “Repetition estimation”. In: *International Journal of Computer Vision (IJCV)*.
- Sakaino, Hidetomo (2008). “Fluid motion estimation method based on physical properties of waves”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Sanchez-Gonzalez, Alvaro, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W Battaglia (2020). “Learning to Simulate Complex Physics with Graph Networks”. In: *arXiv preprint arXiv:2002.09405*.
- Sarel, Bernard and Michal Irani (2005). “Separating transparent layers of repetitive dynamic behaviors”. In: *International Conference on Computer Vision (ICCV)*.

- Scarselli, Franco, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini (2008). “The graph neural network model”. In: *IEEE Transactions on Neural Networks*.
- Schreck, Camille, Christian Hafner, and Chris Wojtan (2019). “Fundamental solutions for water wave animation”. In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.
- Sejdić, Ervin, Yingying Fu, Alison Pak, Jillian A Fairley, and Tom Chau (2012). “The effects of rhythmic sensory cues on the temporal dynamics of human gait”. In: *PloS one*.
- Shelley, Michael J and Jun Zhang (2011). “Flapping and bending bodies interacting with fluid flows”. In: *Annual Review of Fluid Mechanics*.
- Shi, Jianbo and Carlo Tomasi (1994). “Good features to track”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Simonyan, Karen and Andrew Zisserman (2014). “Two-stream convolutional networks for action recognition in videos”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Slaughter, William S (2012). *The linearized theory of elasticity*. Springer Science & Business Media.
- Snoek, Jasper, Hugo Larochelle, and Ryan P Adams (2012). “Practical bayesian optimization of machine learning algorithms”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Spencer, Lisa and Mubarak Shah (2004). “Water video analysis”. In: *International Conference on Image Processing (ICIP)*.
- Sun, Meng, Allan D Jepson, and Eugene Fiume (2003). “Video input driven animation (VIDA)”. In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.
- Taneda, Sadatoshi (1968). “Waving motions of flags”. In: *Journal of the Physical Society of Japan*.
- Thangali, Ashwin and Stan Sclaroff (2005). “Periodic motion detection and estimation via space-time sampling”. In: *Workshops on Application of Computer Vision*.
- Tian, Fang-Bao (2013). “Role of mass on the stability of flag/flags in uniform flow”. In: *Applied Physics Letters*.

- Tokmakov, Pavel, Karteek Alahari, and Cordelia Schmid (2017). "Learning Motion Patterns in Videos". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Torrence, Christopher and Gilbert P Compo (1998). "A practical guide to wavelet analysis". In: *Bulletin of the American Meteorological society*.
- Tralie, Christopher J and Jose A Perea (2018). "(Quasi) Periodicity Quantification in Video Data, Using Topology". In: *Journal on Imaging Sciences*.
- Tran, Du, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri (2018). "A closer look at spatiotemporal convolutions for action recognition". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Truong, Ba Tu and Svetha Venkatesh (2001). "Determining dramatic intensification via flashing lights in movies". In: *International Conference on Multimedia and Expo (ICME)*.
- Tsai, Ping-Sing, Mubarak Shah, Katharine Keiter, and Takis Kasparis (1994). "Cyclic motion detection for motion based recognition". In: *Pattern Recognition*.
- Ummenhofer, Benjamin, Lukas Prantl, Nils Thürey, and Vladlen Koltun (2020). "Lagrangian fluid simulation with continuous convolutions". In: *International Conference on Learning Representations (ICLR)*.
- Van Der Maaten, Laurens (2014). "Accelerating t-SNE using tree-based algorithms". In: *Journal of Machine Learning Research (JMLR)*.
- Van Steenkiste, Sjoerd, Michael Chang, Klaus Greff, and Jürgen Schmidhuber (2018). "Relational neural expectation maximization: Unsupervised discovery of objects and their interactions". In: *International Conference on Learning Representations (ICLR)*.
- Veit, Andreas, Serge Belongie, and Theofanis Karaletsos (2017). "Conditional similarity networks". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Wang, Huamin, James F O'Brien, and Ravi Ramamoorthi (2011). "Data-driven elastic models for cloth: modeling and measurement". In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.

- Wang, Limin, Yuanjun Xiong, Zhe Wang, and Yu Qiao (2015). “Towards good practices for very deep two-stream convnets”. In: *arXiv preprint arXiv:1507.02159*.
- Wang, Tuanfeng Y, Duygu Ceylan, Jovan Popović, and Niloy J Mitra (2019). “Learning a shared shape space for multimodal garment design”. In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.
- Watters, Nicholas, Daniel Zoran, Theophane Weber, Peter Battaglia, Razvan Pascanu, and Andrea Tacchetti (2017). “Visual interaction networks: Learning a physics simulator from video”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Wejchert, Jakub and David Haumann (1991). “Animation aerodynamics”. In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.
- White, Ryan, Keenan Crane, and David A Forsyth (2007). “Capturing and animating occluded cloth”. In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.
- Wu, Hao-Yu, Michael Rubinstein, Eugene Shih, John Guttag, Frédo Durand, and William Freeman (2012). “Eulerian video magnification for revealing subtle changes in the world”. In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.
- Wu, Jiajun, Joseph J Lim, Hongyi Zhang, Joshua B Tenenbaum, and William T Freeman (2016). “Physics 101: Learning Physical Object Properties from Unlabeled Videos”. In: *British Machine Vision Conference (BMVC)*.
- Wu, Jiajun, Erika Lu, Pushmeet Kohli, Bill Freeman, and Josh Tenenbaum (2017). “Learning to see physics via visual de-animation”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Wu, Jiajun, Ilker Yildirim, Joseph J Lim, Bill Freeman, and Josh Tenenbaum (2015). “Galileo: Perceiving physical object properties by integrating a physics engine with deep learning”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Xiao, Jianxiong, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba (2010). “Sun database: Large-scale scene recognition

- from abbey to zoo”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Xue, Tianfan, Jiajun Wu, Zhoutong Zhang, Chengkai Zhang, Joshua B Tenenbaum, and William T Freeman (2018). “Seeing Tree Structure from Vibration”. In: *European Conference on Computer Vision (ECCV)*.
- Yang, Shan, Junbang Liang, and Ming C Lin (2017). “Learning-based cloth material recovery from video”. In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.
- Yang, Shan, Zherong Pan, Tanya Amert, Ke Wang, Licheng Yu, Tamara Berg, and Ming C Lin (2018). “Physics-Inspired Garment Recovery from a Single-View Image”. In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.
- Ye, Tian, Xiaolong Wang, James Davidson, and Abhinav Gupta (2018). “Interpretable intuitive physics model”. In: *European Conference on Computer Vision (ECCV)*.
- Yi, Kexin, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B Tenenbaum (2020). “Clevrer: Collision events for video representation and reasoning”. In: *International Conference on Learning Representations (ICLR)*.
- Zach, C., T. Pock, and H. Bischof (2007). “A Duality Based Approach for Realtime TV-L \pm Optical Flow”. In: *Pattern Recognition*.
- Zhang, Huaidong, Xuemiao Xu, Guoqiang Han, and Shengfeng He (2020). “Context-aware and Scale-insensitive Temporal Repetition Counting”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Zhang, Jun, Stephen Childress, Albert Libchaber, and Michael Shelley (2000). “Flexible filaments in a flowing soap film as a model for one-dimensional flags in a two-dimensional wind”. In: *Nature*.
- Zheng, David, Vinson Luo, Jiajun Wu, and Joshua B Tenenbaum (2018). “Unsupervised learning of latent physical properties using perception-prediction networks”. In: *Conference on Uncertainty in Artificial Intelligence (UAI)*.

LIST OF PUBLICATIONS



This thesis is based on the following publications:

- **Tom F.H. Runia**, Cees G.M. Snoek, and Arnold W.M. Smeulders (2018). “Real-World Repetition Estimation by Div, Grad and Curl”. In: Conference on Computer Vision and Pattern Recognition (CVPR).
- **Tom F.H. Runia**, Cees G.M. Snoek, and Arnold W. M. Smeulders (2019). “Repetition Estimation”. In: International Journal of Computer Vision (IJCV).
- **Tom F.H. Runia**, Kirill Gavriluk, Cees G.M. Snoek, and Arnold W.M. Smeulders (2020). “Cloth in the Wind: A Case Study of Estimating Physical Measurement through Simulation”. In: Conference on Computer Vision and Pattern Recognition (CVPR).
- **Tom F.H. Runia**, Cees G.M. Snoek, and Arnold W. M. Smeulders (2020). “Learning Physical Properties and Relationships from Visual Observations”. Submitted for possible publication.

The author has further contributed to the following publications:

- Rijnder Wever, **Tom F.H. Runia** (2018). “Subitizing with Variational Autoencoders”. In: ECCV Workshops.
- **Tom F.H. Runia**, Cees G.M. Snoek, and Arnold W.M. Smeulders (2018). “Primitive Motion Types for Learning from Instructional Video”. In: CVPR Workshops.
- **Tom F.H. Runia**, Andrew Berneshawi, Rahul Rama Varior, Uta Büchler, Davide Modolo, Joseph Tighe (2019). “Bidirectional GANs for Unsupervised Video Representation Learning”.

SAMENVATTING

SUMMARY IN DUTCH



In dit proefschrift, *Recurrent Motion in Vision*, verkennen we het concept van terugkerende bewegingspatronen in videobeelden. Het proefschrift begint met een studie naar de oorsprong van periodieke bewegingen. Dit leidt tot een nieuwe categorisatie van periodieke beweging in negen fundamentele klassen. In de twee daaropvolgende hoofdstukken introduceren we nieuwe methodes voor het analyseren van terugkerende bewegingen in realistische videos. We vervolgen het proefschrift met de verkenning van herhalende bewegingspatronen in fysische verschijnselen met als doel om materiaaleigenschappen te leren uit het videobeeld.

We dragen het volgende bij aan de *computer vision* literatuur:

Deel I. Recurrent Motion in Vision

- In Hoofdstuk 2 (Runia e.a., 2019) introduceren we een categorisatie van fundamentele periodieke bewegingspatronen. Dit doen we door het 3D bewegingsveld, geïnduceerd door een bewegend object, te ontleden in elementaire differentiaal componenten. Voor de 2D perceptie van 3D periodieke bewegingen in het videobeeld identificeren we twee fundamenteel verschillende cameraposities. Tesamen met de 3D categorisatie resulteert dit in een totaal van 18 observationele gevallen van herhalende beweging.
- In Hoofdstuk 3 (Runia e.a., 2018) introduceren we een praktische oplossing voor het detecteren van herhalende beweging in video. Om een kwalitatieve analyse mogelijk te maken leggen we de nadruk op het tellen van herhalende bewegingen. Onze methode beschouwt de bewegingspatronen uit het vorige hoofdstuk en gebruikt de *continuous wavelet transform* om niet-stationaire herhalende bewegingen te detecteren. Voor de evaluatie introduceren we een nieuwe video dataset met terugkerende bewegingen.
- In Hoofdstuk 4 (Runia e.a., 2019) breiden we de methode van het vorige hoofdstuk verder uit. De nieuwe methode maakt het mogelijk om de terugkerende beweging spatio-temporeel te localiseren in de video. Daarnaast introduceren we een oplossing voor

het tellen van het aantal herhalingen waarbij de waarneming van de beweging significant verandert gedurende de opname, bijvoorbeeld als gevolg van een bewegende camera.

Deel II. Recurrent Physical Dynamics

- In Hoofdstuk 5 (Runia e.a., 2020a) richten we ons op het meten van fysische eigenschappen uit videobeeld zonder het verschijnsel voorheen waargenomen te hebben. De experimenten richten zich op een studie naar vlaggen en doeken in de wind. Onze methode maakt gebruik van simulaties die iteratief worden verbeterd om fysische gelijkenis met het echte videobeeld te realiseren. Daarnaast maken we gebruik van *contrastive learning* en spectrale decompositie van het beeld door middel van neurale netwerken. Om onze methode te evalueren hebben we metingen verricht aan de windsnelheid en bijbehorende opnames van vlaggen in de wind. Dit stelt ons in staat het model te vergelijken met de werkelijkheid.
- In Hoofdstuk 6 beschouwen we het leren van intrinsieke object eigenschappen (zoals massa en restitutiecoëfficiënten) en fysische modellen (zoals veer- en interactiemodellen). Dit doen we door *graph networks* te gebruiken in samenwerking met *contrastive learning* door te leren uit gesimuleerde data. Nadat ons model geleerd heeft van de videobeelden demonsteren we dat het model fysische eigenschappen heeft geleerd alsmede de onderliggende impliciete fysische modellen.

ACKNOWLEDGEMENTS



This PhD could not have been completed without the help of many people to which I would like to express my gratitude. First and foremost, I would like to thank Cees Snoek for allowing me to pursue a doctoral degree in the QUVA Deep Vision lab. I will never forget the job interview we had while I was sitting in a tiny room of my hostel in Bangkok. For me, this initially was a stressful situation as the hostel's internet connection kept dropping out in the hours prior to the call. Fortunately, you were very understanding and this pleasant conversation marked the beginning of our academic relationship. What followed were four years of intense, productive and delightful research with you and Arnold (Smeulders). Cees and Arnold, I am immensely grateful for your supervision, encouragements and scientific advice over the past four years. You have taught me computer vision, helped me to write papers, were always open to my half-baked ideas, attempted to tame my stubbornness, and above all, taught me to think as a scientist. Thank you.

My four years at the University of Amsterdam have been a wonderful time thanks to the endless and enjoyable discussions with colleagues from the Informatics Institute during lunch, meetings and over drinks. In particular, I would like to thank my QUVA lab mates Matthias Reisser, Chang-Yong Oh, Kirill Gavriluk, Noureldien Hussein, Peter O'Connor, Mert Kılıçkaya, Berkay Kıcanaoğlu, Shuai Liao, Maurice Weiler, Adeel Pervez, Amir Ghodrati and Deepak Gupta for the discussions, fun and deadline death-marches.

Beyond my direct lab mates, there are many colleagues and friends from ISIS and AMLab that I would like to thank for making the past years unforgettable: Gjorgji Strezoski, William Thong, Rianne van den Berg, Pascal Mettes, Efstratios Gavves, Thomas Kipf, Bas Veeling, Christos Louizos, Patrick Forré, Maximilian Ilse, Andrew Brown, Nanne van Noord, Devanshu Arya, Jakub Tomczak, Elise van der Pol, Daniel Worrall, Taco Cohen, Karen Ullrich, Thomas Mensink, Herke van Hoof, Sarah Ibrahimi, Inske Groenen, Jiaojiao Zhao, Ivan Sosnovik, Amirhossein Habibi, Mihir Jain, Zenglin Shi, Marcel Worring and many others.

This thesis could not have been completed without the help of Dennis Koelma. Dennis, perhaps you have taught me more than anyone else during the past four years. You were always ready to help with

technical challenges, you have given me the opportunity to run experiments on powerful GPU clusters, and you have immensely expanded my computer science skills — Thank you! Neither would this thesis have been completed without the support from Virginie Mes. Virginie, your help and friendliness over the past four years have been very important, thank you so much! I would also like to thank Mieke van den Berg for her valuable support with research communication.

During the third year of my PhD, I spent four months at Amazon AI in Seattle. I am much obliged to Joe Tighe and Davide Modolo for hosting me at Amazon and also acknowledge Cees for giving the opportunity to reside abroad for some time. I had a wonderful summer in Seattle and would like to thank my friends and colleagues in Seattle for all the hiking, barbecues and conversations. In particular, I would like to thank Daniel McKee, Chunhui Liu, Robin Raw, Andrew Berneshawi, Maartje ter Hoeve, Maxim Berman and Vera von Burg for all the great time we spent in the Pacific Northwest.

I am also grateful for the funding I received from Qualcomm Research and the Advanced School for Computing and Imaging (ASCI). This comfortable financial position allowed me to travel to conferences, participate PhD courses and build a network at summer schools all around the world. All of which have been of great importance to my career.

Outside of academia, I will look back on four beautiful years in Amsterdam. The time in this amazing and vibrant city has been made unforgettable by many friends and family. Specifically, I would like to thank Thomas & Penina, Jip & Anna and Lucie & Caspar for all the wonderful time we have spent together when I was not working in the lab. Additionally, thanks to all my friends in Leiden, Rotterdam, Den Haag and Haarlem for all fun and support over the years.

To my parents, Eelco and Leonoor, thank you for supporting me over all those years. I could not have wished for better life lessons, common values, ethics and scientific preparation during my childhood in preparation of this doctorate degree. Thank you for all the love, advise and incredible support over the past four years.

Finally and most importantly, I would like to thank Guðrun for her endless support throughout this – sometimes stressful – journey. Guðrun, thank you for always being on my side with your emotional support, kindness, love and stressing the importance of being *raskur*. These words will have to do, because no words can truly express my feelings: I love you, and thank you — Tom Runia (Amsterdam, July 5th, 2020)

