Learning from Time

Charig Yang

St Anne's College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Michaelmas 2024

"Of all obstacles to a thoroughly penetrating account of existence, none looms up more dismayingly than 'time'. Explain time? Not without explaining existence. Explain existence? Not without explaining time. To uncover the deep and hidden connection between time and existence, to close on itself our quartet of questions, is a task for the future."

John A. Wheeler

Abstract

With the rise of deep learning, computer vision systems have been highly successful at understanding images. However, understanding the dynamic visual world we live in requires both understanding the appearances of individual image frames, and the temporal relationships between them. This thesis aims to understand videos through the lens of time, by learning from the temporal relationships within image sequences, both instantaneously and over a period of time.

In the first half, we focus on using instantaneous motion – temporal changes between neighbouring video frames – to discover moving objects, based on the intuition that the subject in the video usually moves independently from the background. We propose two methods that can solve this task: first, for a single object in a self-supervised manner by grouping motion into layers, and second, for multiple objects over time in a supervised manner using a vision foundation model. We show applications towards general videos, as well as discovering objects with minimal visibility such as camouflages, where we also present the largest video camouflage dataset to date.

In the second half, we go beyond instantaneous changes and learn from patterns of changes over time, from seconds (natural videos) to days (time-lapse videos) to years (longitudinal images). We leverage the properties of time as a direct supervisory signal, and introduce applications that were previously unachievable in computer vision. We first exploit "uniformity" – that time flows at a constant rate, to read analog clocks in unconstrained scenes. We then relax this constraint to "monotonicity" – that certain changes are consistently unidirectional over a period of time, to discover monotonic changes in a sequence of images. For both cases, we also contribute datasets to foster further research.

Keywords – video understanding, deep learning, motion segmentation, self-supervision

This thesis is submitted to the Department of Engineering Science, The University of Oxford, in fulfilment of the requirements for the degree of Doctor of Philosophy. This thesis is entirely my own work, and except where otherwise stated, describes my own research.

Charig Yang, January 2025.

Acknowledgement

On the previous page, I declared this thesis to be "entirely my own work." While true, that statement feels incomplete, because this thesis is as much about the work as it is about the people who inspired me, supported me, and lifted me up. I am glad to finally have the chance to say thank you.

First, to my supervisors, Andrew Zisserman and Weidi Xie. Thank you for guiding me into the world of computer vision and research – I have never seen anything this beautiful. Thank you for showing me what passion truly means, and for giving me four amazing years of complete freedom to explore, enjoy, and grow – all while providing incredible intellectual and moral support throughout. Not only have you both shaped the researcher I am today, but you also nurtured me as a person: to stay curious, stay humble, and be kind. I will forever be grateful.

A research group is only as great as its people, and my time at VGG has been nothing short of extraordinary. To my collaborators, Hala Lamdouar, Erika Lu, and Junyu Xie, thank you for making the journey joyful, and for helping me grow both as a researcher and a teammate. My thanks go to Guanqi Zhan, Ragav Sachdeva, and Suny Shtedritski for the chats in the lab despite my lack of attendance, Tengda Han and Chuhan Zhang for good food and good life advice, and Laurynas Karazija and Tim Franzmeyer for exploring Seattle together. I also thank Fatma Güney, Shangzhe Wu, Max Bain, Ruth Fong, Lili Momeni, and Andrew Brown for welcoming a clueless undergrad back in the very early days, Ragav Sachdeva, K R Prajwal, and Oishi Deb for pre-deadline paper proofreadings, and Luke Melas-Kyriazi, Nikita Karaev, and Emmanuelle Bourigault for all the memories we shared around the world (both conferences and mountains).

My thanks also extend to Christian Rupprecht and Bill Freeman for an uplifting thesis defense, Hyo Jin Kim alongside the Surreal team at Meta for a lovely time in Seattle, Ashish Thandavan and David Pinto for keeping the GPUs alive, and Wendy Poole for her generous administrative and organizational support throughout.

I am also grateful to the many researchers who continue to inspire me. My journey into computer vision began when I read about Katie Bouman and Bill Freeman's work on photographing a black hole. I learned my first layers from the lectures of Fei-Fei Li and Andrej Karpathy, and I have a special place in my heart for research that contains an immense amount of fun and creativity, including Alyosha Efros's dancing, Bill Freeman's spaghetti, and Carl Vondrick's colorings. I

also carry with me Ross Girshick's advice to find "real tasks" to solve, Lucas Beyer's kind words and encouragement to have fun, and the reminder from Norman Su and David Crandall never to forget an often-overlooked aspect of research: ourselves.

To Dominic Li and Tooki Chu, thank you for growing up with me, for always being there, and for embracing me exactly as I am. I thank Sophie Kuang and Minakshi Ashok for being my emergency contacts, Rose Wei and Terence Tan for counselling and cheesecake, Pang Nganthavee for escape rooms, Lisa Schut for the shared adventures in adulting, and Emily Tsen and El-Amin Ahmed for the steadfast friendship across the years. To my Cohen Quad neighbours, Petra Ferencz, Michael Broome, Adam Milner, Angela Shi, and Maddy Tomlin, thank you for the friendship that extended way beyond that. I also thank Kexin Koh for reminding me to exercise, Aren Karapetyan for the greatest ever hospitality, and Preston Teng, Taiga Kobayashi, Yinni Hu, Nono Sugawara, Helena Watson, Codrutza Dragu, and Kezia Susanto for many, many beers.

I owe appreciation to a special group of people at Oxford – its Thai community. First, to my flatmates, Tai, Ome, Jane and Michiko, for their near-daily companionship and for making our place feel like a true home away from home. I thank Nash for our endless conversations, Pun Pun for teaching me generosity, and Bright and Poon for the "investment" advice. I also thank Petra, Ball, Gunt, Aom, Jean, Ploy, Mos, Ji, and Banky, for the regular kindness and presence, and undergrad friends Byte, Boom, Ou, Techin, Most and Kenneth whose friendships have endured across distance and time. *Khob khun tee rak gun* – I really appreciate it.

I am fortunate enough to be able to pursue my dreams, and I remind myself often of that privilege. I want to express my admiration for those who dedicate themselves to fighting for civil liberties, often at great personal cost. These are the people who inspire me to push the boundaries of what is possible, to dream big and build the future I want to be in, and to never lose sight of what it means to be human – things I will continue to carry with me as I grow.

This thesis may bear only my name, but it is the product of a world of kindness, belief, and shared adventure. Thank you all for being part of it, and above all, my family, who have made this thesis, and everything it represents, possible.

Contents

Chapter 1

Introduction and background

Video is an attractive form of media for many reasons: it is easy to understand, fun to watch, and can make us feel and pay attention more than just pictures or words. Visually, a video is more than just a sequence of images, as the frames are linked across time in a way that makes sense through physics and logic. Beyond entertainment, videos are also used in various domains, including in monitoring of wildlife, nature, healthcare, manufacturing, and surveillance; and navigation of robots and self-driving cars.

Building a computer system to understand videos the way people do is a path towards human-level intelligence. After all, what we see can also be thought of as a video captured with a pair of 'cameras' that are our eyes. Babies also learn certain laws of movement and cause-and-effect relationships by watching people and objects move while experimenting with their interactions with the world around them [Gopnik et al. 2009].

Although computers can understand images with relative ease these days, video understanding has not enjoyed the same level of success. Video is a rich source of information – a movie contains hundreds of thousands of frames, and processing them takes a lot of computing power. More importantly, it is not enough for computers to just "see" each frame, but they also need to understand how the frames connect over time.

This thesis looks at videos through the temporal relationships within image sequences. The thesis is divided into two parts. In the first part, we focus on

discovering what is moving within an image sequence, even if (i) the movement is small, (ii) the subject has low visibility, such as a camouflaged animal, and (iii) the apperance of the subject is unknown. In the second part, we leverage the fundamental rules of time to teach machines to reason about changes over time. We explore the facts that time flows forward at a constant rate ("uniformity"), and that certain changes happen consistently in the same manner over a period of time ("monotonicity").

1.1 Motivation and key ideas

Time is a concept fascinated across many disciplines. In physics, the concept of time is intertwined with the origins of the universe [Hawking 1998], the increase in entropy that provides a direction for time [Boltzmann 1896], and the curvature of spacetime in Einstein's theory of general relativity [Einstein 1922]. In psychology, there have been studies on how humans perceive time and its connection to human memory. In philosophy, time is a widely debated concept, ranging from whether time is a human perception or objective reality, to the ideas of time travel and parallel universes.

This thesis is motivated by these vast understandings and interpretations of time and asks how this can help machines understand the concept of time. This section outlines the key ideas behind the thesis, as well as the relevant high-level background literature.

Figure 1.1: **Object emerges from motion.** In the first two images, we show two neighbouring frames from the series *Squid Game*. The players (in green) were playing a children's game where they get eliminated if they move. The character in front moves only slightly, but the optical flow (right image) clearly highlights the motion. From *Squid Game*, Season 1, Episode 1: *Red Light, Green Light.*

1.1.1 Learning from motion to discover objects

The idea that objects emerge from motion has its roots in psychology. [Wertheimer 1923] introduced the Gestalt principles, among which is the principle of common fate: visual components that move together are perceived as being grouped to the same object. This principle implies that we can separate objects from the background by observing their movement, which explains why hard-to-see objects, like animals in camouflage, become clearly visible once they move. This concept is further supported by the two-stream hypothesis [Goodale and Milner 1992] in the human brain, which suggests that the brain processes visual information through two separate pathways: one for appearance and the other for motion.

In computer vision, the idea of perceptual grouping is closely linked to the task of motion segmentation [Palmer 1999] – obtaining pixel-level masks for moving objects in a sequence. The basic intuition is that objects in motion move independently from their background, and therefore these moving pixels can be grouped to obtain the segmentation for the object.

The question is then, how do we represent motion? While motion is easily explained as what is moving in a sequence of images, its representation takes many forms, from keypoint tracking in humans [S.-E. Wei et al. 2016; L. Ma et al. 2024] to point trajectories [Sand and Teller 2008; Doersch et al. 2022; Harley et al. 2022; Karaev et al. 2024] and dynamic image representations [Bilen et al. 2016]. Between two consecutive frames, motion can be represented as optical flow, which can be thought of as a spatial derivative of the image pixels over time [Horn and Schunck 1981; Lucas and Kanade 1981], as shown in Figure 1.1.

While motion segmentation has existed for a long time [J. Y. Wang and Adelson 1994], it has recently gained popularity due to the advancement of optical flow estimation techniques which enable accurate object boundaries [Teed and Deng 2020]. Apart from segmentation, optical flow is also widely used for different applications from action recognition [Simonyan and Zisserman 2014] to motion magnification [T.-H. Oh et al. 2018].

Figure 1.2: **Continuity error.** Notice in this scene from *Rick and Morty*, the red bags between the two seats disappear for no reason during a conversation. From *Rick and Morty*, Season 2, Episode 2: *Mortynight Run*.

1.1.2 Learning from properties of time

In films and movies, there often exist *continuity errors*, where there are unexplained changes in the scene, due to the oversight in the production. such as those illustrated in Figure 1.2. These mistakes are often very minor, but sometimes bring delight to those with keen eyes (or an algorithm [Pickup and Zisserman 2009]) capable of finding these 'Easter eggs'.

Reflecting upon this, our ability to spot these errors comes from understanding the *continuity* property of time. We understand that things do not suddenly appear or disappear from the world. More generally, time also has other properties that capture the rule of physics and causality, which leads to the predictivity and coherence of events at different timescales.

For instance, we would know if a video is played backwards from our understanding of the arrow of time. Similarly, we would also be able to tell if the ordering of video frames has been shuffled, as it likely disrupts both continuity of motion and logical flow. We also have a good estimation of a reasonable speed based on our intuition of temporal dynamics and physics, and would know if the speed has been tampered.

None of these intuitions can be encoded in static images. Hence, many works have explored using these properties of time for several different applications in video understanding, such as ordering [Misra et al. 2016], speed [Benaim et al. 2020; J. Wang et al. 2020], direction [D. Wei et al. 2018], repetition [Dwibedi et al. 2020], and semantics [Zhukov et al. 2020].

There are two key motivations to do this: first is that video data is abundant

and labelling them at scale is not practical. Hence, learning representations from raw video data itself (also known as "self-supervised learning") is an alternative, and time provides the contextual information for learning the temporal dynamics required for video representations. Second, there are new applications of computer vision that can be uncovered from understanding time, and being able to do new tasks is an interesting direction to explore.

1.1.3 Learning from different video timespans

Learning from natural videos was discussed in the previous two sections. However, due to storage and bandwidth limitations, these videos only capture seconds to minutes of content. There are cases where the temporal change of interest cannot be observed at normal timescales, such as in environmental and wildlife monitoring, medical imaging and construction.

To circumvent this, there exist more long-form videos spanning days to years: time-lapses and longitudinal images. Time-lapse videos involve taking photographs at a low frame rate across a longer duration, so that the resultant sped-up video highlights changes of desired timescale. It can also be made post-hoc by speeding up the pre-recorded video to a higher frame rate. Alternatively, for even longer-term monitoring, the recording may not even be continuous, and instead an image is periodically taken over months or years, such as in satellite images and medical imaging over time.

On the other hand, there also exists slow-motion videos that are captured at very high frame rates over short timespans. This allows for observation of phenomena that are too fast for normal cameras or human eyes to perceive, such as the shattering of glass, fluid dynamics, and fast-moving chemical and biological reactions. These sequences of images can also be considered videos, and understanding what is changing over time is essential.

1.2 Thesis outline and contributions

The main objective of this thesis is to explore novel methods for leveraging temporal signals in videos to improve visual understanding. Specifically, we ask how temporal relationships can be leveraged to uncover structure in the visual world. Our goal is to develop techniques that use these temporal cues to discover and track objects, recognise visual patterns, and solve new applications that are previously unachievable in computer vision.

This thesis is divided into two parts, based on the temporal scale of visual changes being considered. The former concerns instantaneous changes between neighbouring video frames, and the latter considers patterns of changes over time. This section outlines each part of the thesis and its constituent chapters, as well as a summary of the contributions made in each chapter.

Part I: $\frac{\Delta x}{\Delta t}$ – Learning from instantaneous temporal changes

In this half, we consider instantaneous (sub-second) temporal changes in form of motion, and focus on motion segmentation: discovering and segmenting moving objects in a video. We build upon the intuition that salient objects in the video usually move independently from the background.

In Chapter 2, we apply this intuition to a specific task of breaking camouflage, and present a method and dataset for this purpose. In Chapter 3, we move on from a specific use-case to general videos with more complex motion patterns, and directly use this intuition as a self-supervised training objective, allowing motion between video frames to be used as a substitute for manual annotations in performing object discovery. In Chapter 4, we leverage the prowess of a foundation model to obtain high-quality segmentation masks, as well as extend the method from frame-wise segmentation to connecting the segmentation masks across frames throughout a video sequence.

Part II: $x(t)$ – Learning from change patterns over time

In this half, we explore longer-term temporal changes from seconds to years, and focus on exploring new applications of computer vision that were previously unattainable in the literature. We present two distinct papers building upon different intuitions based on properties of time, each with new dataset contributions.

First, in Chapter 5, we explore reading analog clocks in unconstrained scenes without manual supervision. We use the intuition that time flows at a constant rate to generate reliable pseudo-labels from unlabelled clock videos. Second, in Chapter 6, we explore discovering changes that are monotonic over the period of time. This is achieved via a simple self-supervised task of ordering a shuffled sequence, by exploiting the intuition that only monotonic changes can give rise to the correct ordering.

1.3 Publications

Chapters 2 to 6 each contains a research paper that has been peer-reviewed and accepted for publication at an international conference.

Chapter 2: Hala Lamdouar, **Charig Yang**, Weidi Xie, and Andrew Zisserman. Betrayed by Motion: Camouflaged object discovery via motion segmentation. In Proceedings of the Asian Conference on Computer Vision, 2020.

Chapter 3: **Charig Yang**, Hala Lamdouar, Erika Lu, Andrew Zisserman, and Weidi Xie. Self-supervised video object segmentation by motion grouping. In Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021. A short version of this paper also won the Best Paper Award at CVPR Workshop, 2021.

Chapter 4: Junyu Xie, **Charig Yang**, Weidi Xie, and Andrew Zisserman. Moving Object Segmentation: All you need is SAM (and flow). In Proceedings of the Asian Conference on Computer Vision, 2024 (Oral Presentation).

Chapter 5: **Charig Yang**, Weidi Xie, and Andrew Zisserman. It's About Time: Analog clock reading in the wild. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022.

Chapter 6: **Charig Yang**, Weidi Xie, and Andrew Zisserman. Made to Order: Discovering monotonic temporal changes via self-supervised video ordering. In Proceedings of the European Conference on Computer Vision, 2024 (Oral Presentation).

Publications not included

Charig Yang, Samiul Alam, Shakhrul Iman Siam, Michael J. Proulx, Lambert Mathias, Kiran Somasundaram, Luis Pesqueira, James Fort, Sheroze Sheriffdeen, Omkar Parkhi, Yuheng Ren, Mi Zhang, Yuning Chai, Richard Newcombe, and Hyo Jin Kim. Reading Recognition in the Wild. In Proceedings of the International Conference on Neural Information Processing Systems, 2025.

Part I

$\frac{\Delta x}{\Delta t}$ – Learning from instantaneous temporal changes

Chapter 2

Betrayed by Motion: Camouflaged object discovery via motion segmentation

The paper was published at the Asian Conference on Computer Vision, 2020.

time flies

Generated with Meta AI

Betrayed by Motion: Camouflaged object discovery via motion segmentation

Hala Lamdouar Charig Yang Weidi Xie Andrew Zisserman

VGG, Department of Engineering Science, University of Oxford

{lamdouar,charig,weidi,az}@robots.ox.ac.uk

http://www.robots.ox.ac.uk/~vgg/data/MoCA

Abstract

The objective of this paper is to design a computational architecture that discovers camouflaged objects in videos, specifically by exploiting motion information to perform object segmentation. We make the following three contributions: (i) We propose a novel architecture that consists of two essential components for breaking camouflage, namely, a differentiable registration module to align consecutive frames based on the background, which effectively emphasises the object boundary in the difference image, and a motion segmentation module with memory that discovers the moving objects, while maintaining the object permanence even when motion is absent at some point. (ii) We collect the first large-scale Moving Camouflaged Animals (MoCA) video dataset, which consists of over 140 clips across a diverse range of animals (67 categories). (iii) We demonstrate the effectiveness of the proposed model on MoCA, and achieve competitive performance on the unsupervised segmentation protocol on DAVIS2016 by only relying on motion.

2.1 Introduction

We consider a fun yet challenging problem of breaking animal camouflage by exploiting their motion. Thanks to years of evolution, animals have developed the ability to hide themselves in the surrounding environment to prevent being noticed by their prey or predators. Consider the example in Figure 2.1a, discovering the fish by its appearance can sometimes be extremely challenging, as the animal's texture is indistinguishable from its background environment. However, when the fish starts moving, even very subtly, it becomes apparent from the motion, as shown in Figure 2.1c. Having the ability to segment objects both in still images, where this is possible, and also from motion, matches well to the two-stream hypothesis in neuroscience. This hypothesis suggests that the human visual cortex consists of two different processing streams: the ventral stream that performs recognition, and the dorsal stream that processes motion [Goodale and Milner 1992], providing strong cues for visual attention and detecting salient objects in the scene.

(a) t-th frame. (b) (t+1)-th frame. (c) Optical flow.

Figure 2.1: Two consecutive frames from the camouflage dataset, with a bounding box denoting the salient object. When the object starts moving, even subtly, we are able to detect it more easily, as shown, in the computed optical flow.

In recent years, computer vision research has witnessed tremendous progress, mainly driven by the ability of learning effective representations for detecting, segmenting and classifying objects in *still images*. However, the assumption that objects can be well-segmented by their appearance alone is clearly an oversimplification; that is to say, if we draw an analogy from the two-stream hypothesis, the computer vision systems trained on images can only mimic the function of the ventral stream. The goal of this paper is to develop a computational architecture that is able to process motion representations for breaking camouflage, *e.g.* by taking optical flow (Figure 2.1c) as input, and predicting a segmentation mask for the animal of interest.

Unfortunately, simply relying on motion will not solve our problem completely,

as, *first*, optical flow estimation itself remains extremely challenging and is under active research. In practice, modern optical flow estimation techniques provide a fairly good indication of rough object motion, but not fine-grained details, *e.g.* the exact shape of the objects and their contours. To compensate for the missing details, we propose to use a differentiable registration module for aligning consecutive frames, and use the difference of the registered images as auxiliary information to determine the exact contour; *Second*, if the motion stops at certain points in the video sequence, then a memory module is required to maintain the object permanence, as is done in [Tokmakov et al. 2019], *i.e.* to capture the idea that objects continue to exist even when they cannot be seen explicitly in the motion representation.

Another main obstacle encountered when addressing the challenging task of camouflage breaking is the limited availability of benchmarks to measure progress. In the literature, there is a Camouflaged Animals video dataset, released by Bideau *et al.* [Bideau and Learned-Miller 2016a], but this has only 9 clips on 6 different kinds of animals (about 840 frames). To overcome this limitation, we collect a **Mo**ving **C**amouflaged **A**nimal dataset, termed **MoCA**, which consists of 141 video sequences (37K frames), depicting 67 kinds of camouflaged animals moving in natural scenes. Both temporal and spatial annotations are provided in the form of tight bounding boxes on every 5th frame and the rest are linearly interpolated.

To summarize, in this paper, we make the following contributions: *First*, we propose a novel architecture with two essential components for breaking camouflage, namely, a differentiable registration module to align the background (regions other than the camouflaged animal) of consecutive frames, which effectively highlights the object boundary in the difference image, and a motion segmentation module with memory that discovers moving objects, while maintaining the object permanence even when motion is absent at some point. *Second*, we collect the first large-scale video camouflage benchmark (MoCA), which we release to the community for measuring progress on camouflage breaking and object tracking. *Third*, we demonstrate the effectiveness of the proposed model on MoCA, outperforming the previous video segmentation approaches using motion. In addition, we also benchmark on DAVIS2016, achieving competitive performance on the unsupervised segmentation protocol despite using only motion. Note that, DAVIS2016 is

fundamentally different from MoCA, in the sense that the objects are visually distinctive from the background, and hence motion may not be the most informative cue for segmentation.

2.2 Related work

Our work cuts across several areas of research with a rich literature, we can only afford to review some of them here.

Video object segmentation [Bideau and Learned-Miller 2016a; Pont-Tuset et al. 2017; Xu et al. 2018; Brox and Malik 2010; Ochs and Brox 2011; Papazoglou and Ferrari 2013; Jain et al. 2017; Tokmakov et al. 2019; Dave et al. 2019; S. W. Oh et al. 2019; Voigtlaender et al. 2019; Vondrick et al. 2018; W. Wang et al. 2019; Lai and W. Xie 2019; Lai et al. 2020; Maninis et al. 2018; Voigtlaender and Leibe 2017; Caelles et al. 2017; Fragkiadaki et al. 2012; Keuper et al. 2015; Z. Yang et al. 2019; X. Lu et al. 2019] refers to the task of localizing objects in videos with pixel-wise masks. In general, two protocols have recently attracted an increasing interest from the vision community [Pont-Tuset et al. 2017; Xu et al. 2018], namely unsupervised video object segmentation (**unsupervised VOS**), and semi-supervised video object segmentation (**semi-supervised VOS**). The former aims to automatically separate the object of interest (usually the most salient one) from its background in a video sequence; and the latter aims to re-localize one or multiple targets that are specified in the first frame of a video with pixel-wise masks. The popular methods to address the unsupervised VOS have extensively relied on a combination of appearance and motion cues, *e.g.* by clustering trajectories [Brox and Malik 2010; Ochs and Brox 2011], or by using two stream networks [Papazoglou and Ferrari 2013; Jain et al. 2017; Tokmakov et al. 2019; Dave et al. 2019]; or have purely used appearance [W. Wang et al. 2019; Z. Yang et al. 2019; Jun Koh and C.-S. Kim 2017; Fan et al. 2019]. For semi-supervised VOS, prior works can roughly be divided into two categories, one is based on mask propagation [S. W. Oh et al. 2019; Voigtlaender et al. 2019; Vondrick et al. 2018; W. Wang et al. 2019; Lai and W. Xie 2019; Lai et al. 2020], and the other is related to few shot learning or online adaptation [Maninis et al. 2018; Voigtlaender and Leibe 2017;

Caelles et al. 2017].

Camouflage breaking [Bideau and Learned-Miller 2016a; Le et al. 2016] is closely related to the unsupervised VOS, however, it poses an extra challenge, as the object's appearance from *still image* can rarely provide any evidence for segmentation, *e.g.* boundaries. As such, the objects or animals will only be apparent when they start to move. In this paper, we are specifically interested in this type of problem, *i.e.* breaking the camouflage in a class-agnostic manner, where the model takes no prior knowledge of the object's category, shape or location, and is asked to discover the animal with pixel-wise segmentation masks whenever they move.

Image registration/alignment is a long-standing vision problem with the goal of transferring one image to another with as many pixels in correspondence as possible. It has been applied to numerous applications such as video stabilization, summarization, and the creation of panoramic mosaics. A comprehensive review can be found in [Hartley and Zisserman 2000; Szeliski 2004]. In general, the pipeline usually involves both correspondence estimation and transformation estimation. Traditionally, the alignment methods apply hand-crafted features, *e.g.* SIFT [Lowe 1999], for keypoint detection and matching in a pair of images, and then compute the transformation matrix by solving a linear system. To increase the robustness of the geometric transformation estimation, RANdom SAmple Consensus (RANSAC) [Fischler and Bolles 1981] is often adopted. In the deep learning era, researchers have constructed differentiable architectures that enable end-to-end optimization for the entire pipeline. For instance, [Brachmann et al. 2017] proposed a differentiable RANSAC by relaxing the sparse selection with a soft-argmax operation. Another idea is to train a network with binary classifications on the inliers/outliers correspondences using either ground truth supervision or a soft inlier count loss, as in [Brachmann and Rother 2018; Ranftl and Koltun 2018; Rocco et al. 2018; Brachmann and Rother 2019], and solve the linear system with weighted least squares.

2.3 A video registration and segmentation network

At a high level, we propose a novel computational architecture for breaking animal camouflage, which *only* considers motion representation as input, *e.g.* optical flow, and produces the segmentation mask for the moving objects. Specifically, as shown in Figure 2.2, the model consists of two modules: (i) a differentiable registration module for aligning consecutive frames, and computing the *difference* image to highlight the fine-grained detail of the moving objects, *e.g.* contours and boundaries (Section 2.3.2); and (ii) a motion segmentation network, which takes optical flow together with the difference image from the registration as input, and produces a detailed segmentation mask (Section 2.3.3).

2.3.1 Motion representation

In this paper, we utilise optical flow as a representation of motion. Formally, consider two frames in a video sequence, I_t and I_{t+1}, each of dimension $\mathbb{R}^{H\times W\times 3}$, the optical flow is defined as the displacement field $F_{t\to t+1} \in \mathbb{R}^{H\times W\times 2}$ that maps each pixel from I_t to a corresponding one in I_{t+1}, such that:

$$I_t(\mathbf{x}) = I_{t+1}(\mathbf{x} + F_{t\to t+1}(\mathbf{x})), \tag{2.1}$$

where $\mathbf{x}$ represents the spatial coordinates (x, y) and F represents the vector flow field in both horizontal and vertical directions. In practice, we use the pretrained PWCNet [D. Sun et al. 2018] for flow estimation, some qualitative examples can be found in Figure 2.1 and Figure 2.5.

2.3.2 Differentiable registration module

One of the main challenges of segmentation with optical flow is the loss of rich fine-grained details due to motion approximations. In order to recover the sharp contours of the objects under motion, we seek a low level RGB signal which ideally suppresses the background and highlights the object's boundaries. In this paper, we use the image difference between consecutive frames after camera motion compensation. To this end, a reasonable assumption to make is that the

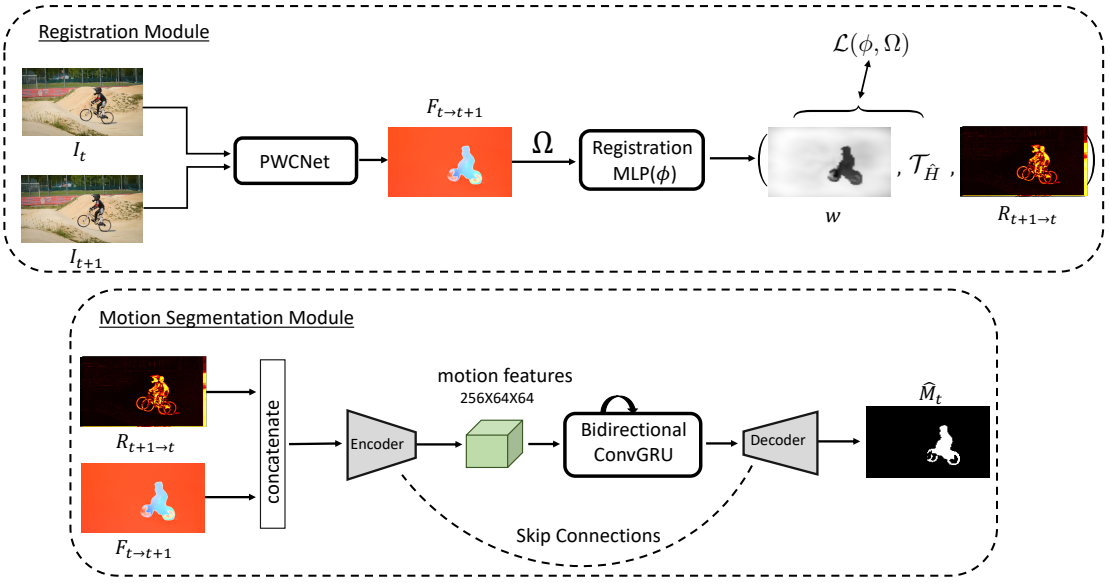


Figure 2.2: Architecture Overview. The proposed architecture is composed of two different modules, namely registration and motion segmentation.

foreground object undergoes an independent motion with respect to the global transformation of the background. We propose a differentiable registration module for estimating this transformation, which we approximate with a homography ($\mathcal{T}_{\hat{H}}$) between consecutive frames, and then compute the *difference* image after alignment ($R_{t+1\to t} = |\mathcal{T}_{\hat{H}}(I_{t+1}) - I_t|$), which will provide cues for the animal contours.

The key here is to train a registration module that accepts a set of correspondences obtained from the consecutive frames, outputs an homography transformation matrix (H), and an inlier weight map (w), which, ideally, acts like a RANSAC process, and has 1's for every background pixel, and 0's for every foreground pixel (moving object's). In this paper, we parametrize the registration module with Multiple Layer Perceptrons (MLPs), *i.e.* $[H, w] = \phi(\{p_s, p_t\}; \theta_r)$, where $p_s \in \mathcal{R}^{mn\times 2}$ denotes the spatial coordinates of all pixels (normalized within the range $[-1, 1]$) in the source image, and their corresponding position in the target image ($p_t \in \mathcal{R}^{mn\times 2}$), based on the computed optical flow, and θ_r are the trainable parameters.

Homography transformation.

In order to be self-contained, we summarise here the homography computation. Mathematically, a homography transformation ($\mathcal{T}_H$) maps a subset of points $\mathcal{S}_s$ from the source image to a subset of points $\mathcal{S}_t$ in the target image; in our case, the source and target images refer to I_{t+1} and I_t respectively:

$$\forall {p_i}^s \in \mathcal{S}_s,\ \exists {p_i}^t \in \mathcal{S}_t \quad {p_i}^t = \mathcal{T}_H({p_i}^s) = \alpha_i H {p_i}^s, \tag{2.2}$$

where H is the matrix associated with the homography transformation $\mathcal{T}_H$ with 8 degrees of freedom, and α_i a non-zero scalar. This formulation can be expressed using homogeneous coordinates of ${p_i}^s$ and ${p_i}^t$ as:

$$\begin{pmatrix} {x_i}^t \\ {y_i}^t \\ 1 \end{pmatrix} = \alpha_i H \begin{pmatrix} {x_i}^s \\ {y_i}^s \\ 1 \end{pmatrix}. \tag{2.3}$$

Using the standard Direct Linear Transform (DLT) [Hartley and Zisserman 2000], the previous equation can be written as:

$$A\ vec(H) = \mathbf{0}, \tag{2.4}$$

where $vec(H) = \begin{pmatrix} h_{11} & h_{12} & h_{13} & h_{21} & h_{22} & h_{23} & h_{31} & h_{32} & h_{33} \end{pmatrix}^T$ is the vectorised homography matrix and A the data matrix. The homography H can therefore be estimated by solving such over-complete linear equation system. For more details on the DLT computation, refer to the Appendix.

Training objective

In order to train the registration module ($\phi(\cdot;\theta_r)$), we can optimize:

$$\mathcal{L} = \frac{1}{\sum w} \sum_{\Omega} w \cdot ||\mathcal{T}_H(p_s) - p_t||_2 + R(w), \tag{2.5}$$

where Ω refers to all the pixels on the mn grid. Note that, the homography $\mathcal{T}_H$ transformation in this case can be solved with a simple weighted least square (WLS) and differentiable SVD [Ranftl and Koltun 2018] for parameter updating. To avoid trivial solution, where the weight map can be full of zeros that perfectly minimize the loss, we add a regularization term ($R(w)$), that effectively encourages as many inliers as possible:

$$R(w) = -\gamma \sum_{p \in \Omega} l_p - \frac{1}{mn} \sum_{\Omega} \Big\{ l_p \cdot \log(w) + (1 - l_p) \cdot \log(1 - w) \Big\}, \tag{2.6}$$

$$\text{where} \quad l_p = \sigma\{(\epsilon - ||\mathcal{T}_H(p_s) - p_t||_2)/\tau\}.$$

In our training, $\gamma = 0.05$, $\tau = 0.01$, $\epsilon = 0.01$ and $m = n = 64$. The first term in $R(w)$ (l_p) refers to a differentiable inlier counting [Brachmann and Rother 2018; Rocco et al. 2018]. The rest of the terms aim to minimize the binary cross-entropy at each location of the inlier map, as in [Moo Yi et al. 2018], forcing the predictions to be classified as inlier (1's) or outlier (0's).

2.3.3 Motion segmentation module

After introducing the motion representation and registration, we consider a sequence of frames from the video, $I \in \mathcal{R}^{T\times 3\times H\times W}$, where the three channels refer to a concatenation of the flow (2 channels) and difference image (1 channel). For simplicity, here we use a variant of UNet [Ronneberger et al. 2015] with the bottleneck feature maps being recurrently processed.

Specifically, the **Encoder** of the segmentation module will independently process the current inputs, ending up with motion features of $\mathcal{R}^{T\times 256\times 64\times 64}$, where $T, 256, 64, 64$ refer to the number of frames, number of channels, height, and width respectively. After the Encoder, the **memory module** (a bidirectional convGRU [Ballas et al. 2016] is used in our case) operates on the motion features, updating them by aggregating the information from time steps in both directions. The **Decoder** takes the updated motion features, and produces an output binary segmentation mask, *i.e.* foreground vs background. The Motion Segmentation Module is trained with pixelwise binary cross-entropy loss.

This completes the descriptions of the two individual modules used in the proposed architectures. Note that the entire model is trained together as it is end-to-end differentiable.

2.4 MoCA: a new Moving Camouflaged Animal dataset

One of the main obstacles encountered when addressing the challenging task of camouflage breaking is the limited availability of datasets. A comparison of existing datasets in Table 2.1. Bideau *et al.* published the first Camouflaged Animals video dataset with only 9 clips, and Le *et al.* proposed the CAMO dataset with

Datasets	# Clips	# Images	# Animals	Video
Camouflaged Animals	9	839	6	✓
CAMO	0	1250	80	×
MoCA (Ours)	141	37K	67	✓

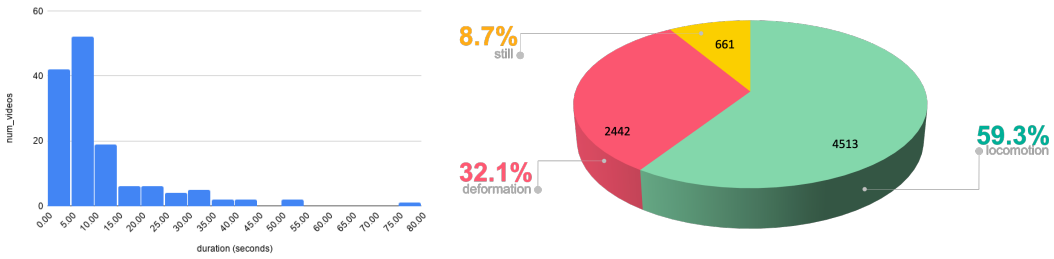
Table 2.1: Statistics for recent camouflage breaking datasets; "# Clips" denotes the number of clips in the dataset; "# Images" denotes the number of frames or images in the dataset; "# Animals" denotes the number of different animal categories in the dataset; "Video" indicates whether videos are available.

only single image camouflage, and therefore not suitable for our video motion segmentation problem.

To overcome this limitation, we collect the **Mo**ving **C**amouflaged **A**nimal dataset, termed **MoCA**, which consists of 141 video sequences depicting various camouflaged animals moving in natural scenes. We include the PWC-Net optical flow for each frame and provide both spatial annotations in the form of bounding boxes and motion labels every 5-th frame with the rest linearly interpolated.

2.4.1 Detailed statistics

MoCA contains 141 video sequences collected from YouTube, at the highest available resolution, mainly 720×1280, and sampled at $24 fps$. A total of 37250 frames spanning 26 minutes. Each video represents a continuous sequence depicting one camouflaged animal, ranging from 1.0 to 79.0 seconds. The distribution of the



(a) Distribution of video lengths. (b) Distribution of motion labels.

Figure 2.3: Statistics for the Moving Camouflaged Animals (MoCA) dataset. In (a), the x-axis denotes the video duration, and the y-axis denotes the number of video sequences. (b) the distribution of frames according to their motion types (still, deformation and locomotion), see text for the detailed definitions.

video lengths is shown in Figure 2.3a. The dataset is labelled with a bounding box in each frame, as well as a motion label for the type of motion. While annotating the data, we distinguish three types of motion:

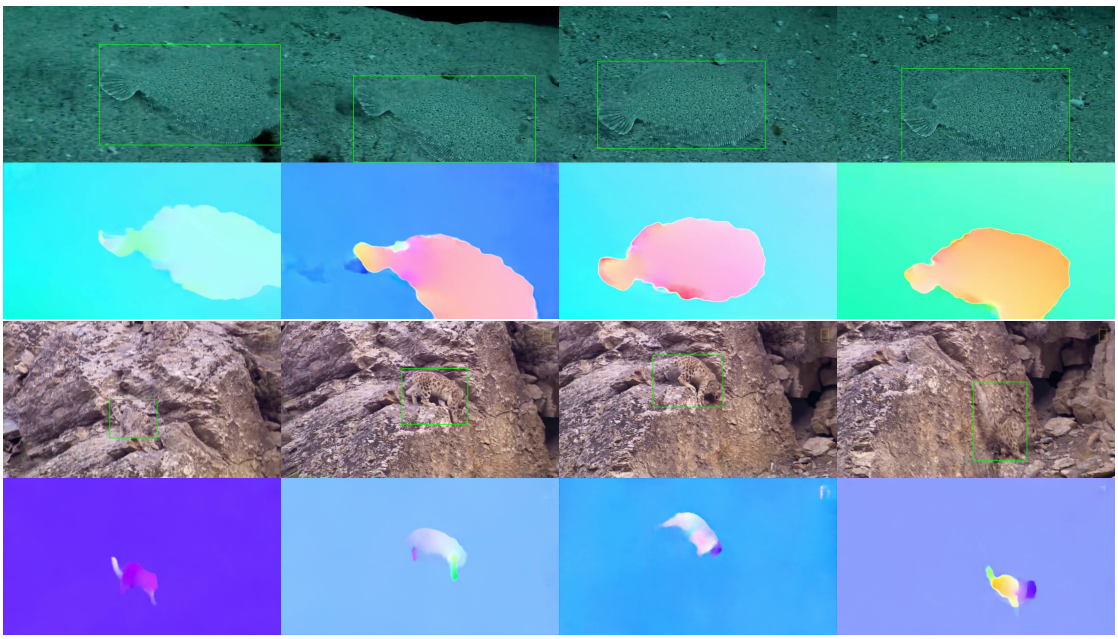


Figure 2.4: Example sequences from the Moving Camouflaged Animals (MoCA) dataset with their corresponding optical flows.

- **Locomotion:** when the animal engages in a movement that leads to a significant change of its location within the scene *e.g.* walking, running, climbing, flying, crawling, slithering, swimming, *etc.*

- **Deformation:** when the animal engages in a more delicate movement that only leads to a change in its pose while remaining in the same location *e.g.* moving a part of its body.

- **Static:** when the animal remains still.

As shown in Figure 2.3b, motion wise, the dataset contains 59.3% locomotion, 32.1% deformations, and 8.7% still frames. Examples from the dataset are shown in Figure 2.4.

2.5 Experiments

In this section, we detail the experimental setting used in this paper, including the datasets, evaluation metrics, baseline approaches and training details.

2.5.1 Datasets

DAVIS2016 refers to the Densely Annotated VIdeo Segmentation dataset [Perazzi et al. 2016]. It consists of 50 sequences, 30 for training and 20 for testing, captured at $24fps$ and provided at two resolutions. We use the $480p$ version in all experiments. This dataset has the advantage that accurate pixelwise ground truth segmentations are provided, $3,455$ annotations in total, as well as spanning a variety of challenges, such as occlusions, fast-motion, non-linear deformation and motion-blur. We train the model on the DAVIS 2016 training set and report results on its validation splits.

Synthetic Moving Chairs. In order to train the differentiable registration module properly, we use video sequences that are synthetically generated. Specifically, we use the 3D-rendered objects from the Flying Chairs dataset as foreground, and take random images from YouTube-VOS as background. We then apply rigid motions, *e.g.* homographies, to the background, and simulate an independent motion for the foreground object. Note that, with such a synthetic dataset, we have complete information about the background homography transformation, optical flow, inlier maps, and object masks, which enables us to better initialise the registration module before training on real video sequences. These synthetic video sequences were only used to pre-train the registration and motion segmentation modules. We include example images in the Appendix.

Evaluation metrics. Depending on the benchmark dataset used, we consider two different evaluation metrics. For **DAVIS2016**, we follow the standard protocol for unsupervised video object segmentation proposed in [Perazzi et al. 2016], namely, the mean region similarity $\mathcal{J}$, which is the intersection-over-union (IOU) of the prediction and ground truth; and mean contour accuracy $\mathcal{F}$, which is the F-measure defined on contour points from the prediction and the ground truth. For **MoCA**, as we only have the bounding box annotations, we define the metric as the IOU between the ground truth box and the minimum box that includes the predicted segmentation mask. Note that, we follow the same protocol used in Bideau *et al.* [Bideau and Learned-Miller 2016a], meaning, we only evaluate the segmentation of the animals under locomotion or deformation (*not* in static

frames).

2.5.2 Baselines

We compare with five previous state-of-the-art approaches [Tokmakov et al. 2017; Tokmakov et al. 2019; Dave et al. 2019; Z. Yang et al. 2019; X. Lu et al. 2019]. In [Tokmakov et al. 2019], Tokmakov *et al.*, the LVO method uses a two-stream network for motion segmentation, where the motion stream accepts optical flow as input via a MPNet [Tokmakov et al. 2017] architecture, and the appearance stream uses an RGB image as input. Similar to ours, a memory module is also applied to recurrently process the frames. A more recent approach [Dave et al. 2019] adapts the Mask-RCNN architecture to motion segmentation, by leveraging motion cues from optical flow as a bottom-up signal for separating objects from each other, and combines this with appearance evidence for capturing the full objects. For fair comparison across methods, we use the same optical flow computed from PWCNet for all the flow-based methods. To this end, we re-implement the original MPNet [Tokmakov et al. 2017], and train on the synthetic FT3D dataset [Mayer et al. 2016]. For LVO [Tokmakov et al. 2019], we also re-train the pipeline on DAVIS2016. For Seg-det [Dave et al. 2019] and Seg-track [Dave et al. 2019], we directly replace the flows from Flownet2 [Ilg et al. 2017] with the ones from PWC-Net in the model provided by the authors. In all cases, our re-implemented models outperform or match the performance reported in the original papers. In addition, we also compare our method to AnchorDiff [Z. Yang et al. 2019] and COSNet [X. Lu et al. 2019], both approaches have been trained for unsupervised video object segmentation with only RGB video clips, and show very strong performance on DAVIS.

2.5.3 Training and architecture details

Here we describe the main modules of our pipeline. More details can be found in the Appendix.

Registration module. We adopt the architecture of [Moo Yi et al. 2018], which is MLPs with 12 layer residual blocks. We first train the registration module on the Synthetic Moving Chairs sequences described in 2.5.1, for $10K$ iterations

using an Adam optimizer with a weight decay of 0.005 and a batch size of 4. For a more stable training, we use a lower learning rate, *i.e.* 5×10^{-5}, avoiding the ill-conditioned matrix in the SVD.

Motion segmentation module. We adopt a randomly initialized ResNet-18. Frame-wise segmentation is trained from scratch on the synthetic dataset, together with the pre-trained registration module. We further include the bidirectional ConvGRU and finetune the whole pipeline on DAVIS 2016, with each sequence of length 11, batch size of 2, for a total of $25K$ iterations. For all training experiments, we use frames with a resolution of $\mathcal{R}^{256\times256\times3}$.

2.6 Results

In this section, we first describe the performance of our model and previous state-of-the-art approaches on the new MoCA dataset, and then compare segmentation performance on the DAVIS2016 benchmark.

Input	Model	RGB	Flow	Register	Memory	Locomotion	Deform	All_Motion
Flow	MPNet [Tokmakov et al. 2017]	×	✓	×	✓	21.3	**23.5**	22.2
	ours-A	×	✓	×	✓	31.3	17.8	27.6
	ours-B	×	✓	MLP	×	29.9	15.6	25.5
	ours-C	×	✓	MLP	✓	**47.8**	20.7	**39.4**
	ours-D	×	✓	RANSAC	✓	42.9	19.2	35.8
RGB	AnchorDiff [Z. Yang et al. 2019]	✓	×	×	×	30.9	29.4	30.4
	COSNet [X. Lu et al. 2019]	✓	×	×	×	35.9	35.1	36.2
Both	LVO [Tokmakov et al. 2019]	✓	✓	×	✓	30.6	34.9	30.6
	Seg-det [Dave et al. 2019]	✓	✓	×	×	16.9	18.7	17.9
	Seg-track [Dave et al. 2019]	✓	✓	×	×	29.9	32.2	30.2
	ours-E	✓	✓	MLP	✓	**45.0**	**38.0**	**42.4**

Table 2.2: Mean Intersection Over Union on MoCA for the different motion type subsets. "All_Motion" refers to the overall performance

2.6.1 Results on the MoCA benchmark

We summarise all the quantitative results in Table 2.2 and discuss them in the following sections. In Figure 2.5, we illustrate the effect of the differentiable registration and Figure 2.6 shows examples of the overall segmentation method. More examples are presented in the Appendix.

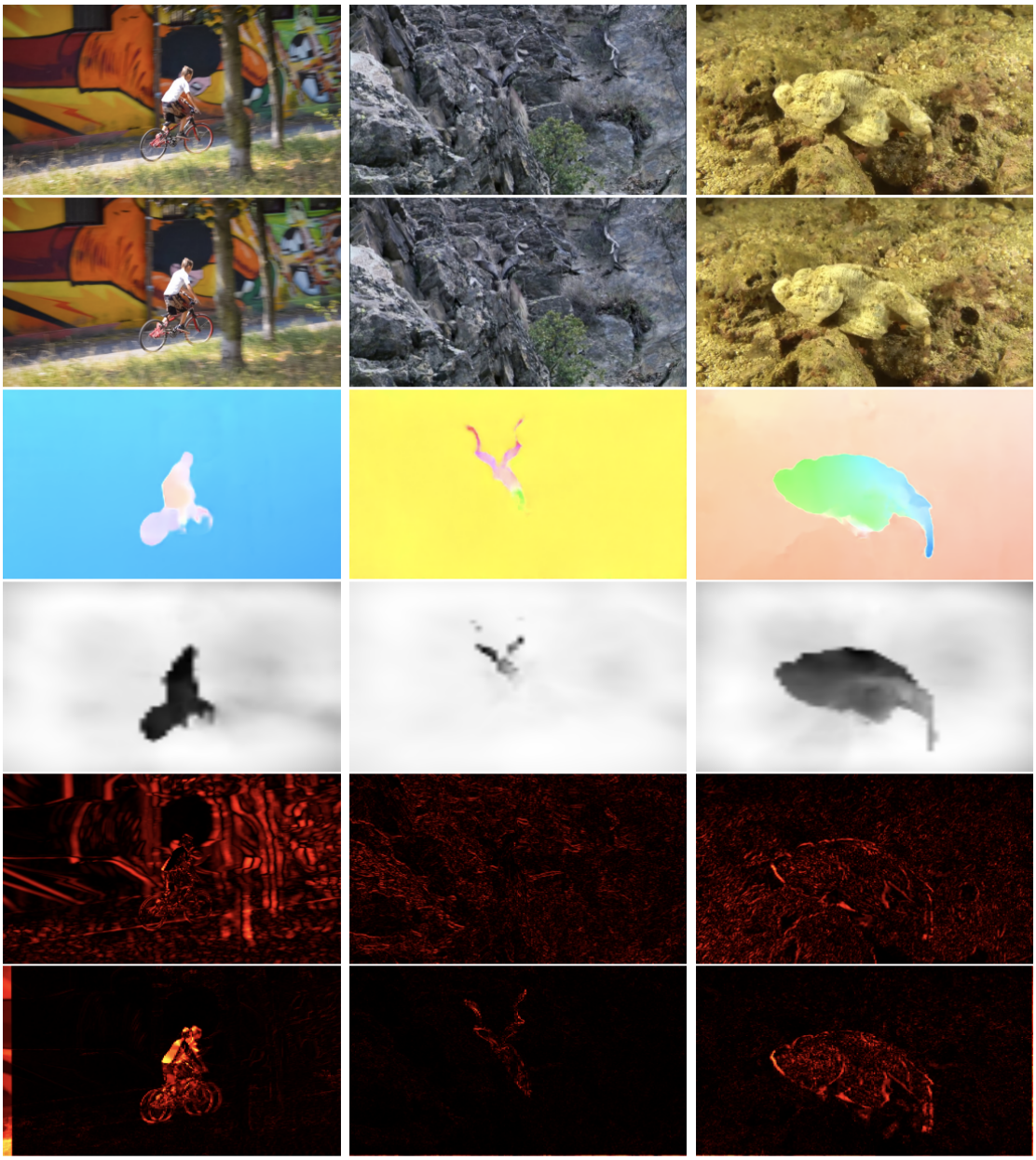
Effectiveness of registration/alignment. To demonstrate the usefulness of the differentiable registration module, we carry out an ablation study. By

comparing ours-A (without registration), ours-D (registration using RANSAC) and Ours-C (registration using our trainable MLPs), it is clear that the model with trainable MLPs for alignment helps to improve both the animal discovery on video sequences with locomotion and deformation, outperforming ours-A (without registration) and ours-D (RANSAC).

In Figure 2.5 we visualise the results from the registration module, *e.g.* the inlier map, difference image before alignment (second last row) and after alignment (last row). It is clear that the difference images computed after alignment are able to ignore the background, and successfully highlight the boundary of the moving objects, *e.g.* the wheels of the bicycle from the first column.

Effectiveness of memory. Comparing model-B (without memory) and model-C (with memory), the only difference lies on whether the frames are processed individually or recurrently. As shown by the results, a significant boost is obtained with the help of the memory module (25.5 vs. 39.4 on All_Motion), showing its effectiveness.

Comparison to baselines and previous approaches. From Table 2.2, we make the following observations: *First*, when comparing with MPNet [Tokmakov et al. 2017], which also processes optical flow, our model demonstrates a superior performance on All_Motion (39.4 vs 22.2), and an even larger gap on Locomotion, showing the usefulness of image registration and memory modules; *Second*, as expected, the state-of-the-art unsupervised video segmentation approaches relying on appearance (RGB image as input), *e.g.* AnchorDiff [Z. Yang et al. 2019] and COSNet [X. Lu et al. 2019], tend to struggle on this camouflage breaking task, as the animals often blend with the background, and appearance then does not provide informative cues for segmentation, emphasising the importance of motion information (by design) in this dataset; *Third*, when we adopt a two-stream model, *i.e.* extend the architecture with a Deeplabv3-based appearance model, and naively average the prediction from appearance and flow models, the performance can be further boosted from 39.4 to 42.4, significantly outperforming all the other two-stream competitors, *e.g.* LVO [Tokmakov et al. 2019], Seg-det and Seg-seg [Dave et al. 2019].



(a) (b) (c)

Figure 2.5: Registration results from (a) the validation set of DAVIS 2016; and (b)(c) MoCA. From top to bottom: Frame t, Frame $t+1$, Forward PWCNet optical flow, inlier weights (background pixels are shown as gray or white), image difference without alignment, aligned image difference.

2.6.2 Results on DAVIS2016 benchmark

We compare to previous approaches on the Unsupervised Video Object Segmentation protocol. In all experiments, we *do not* use any post-processing *e.g.* CRF. Both our re-trained MPNet and LVO model outperform the original reported results in the papers, which guarantees a fair comparison with their model. For Seg-det, replacing the PWC-Flow leads to a small drop (around 2.3%) in the mean $\mathcal{J}$, but this will not affect our conclusion here.

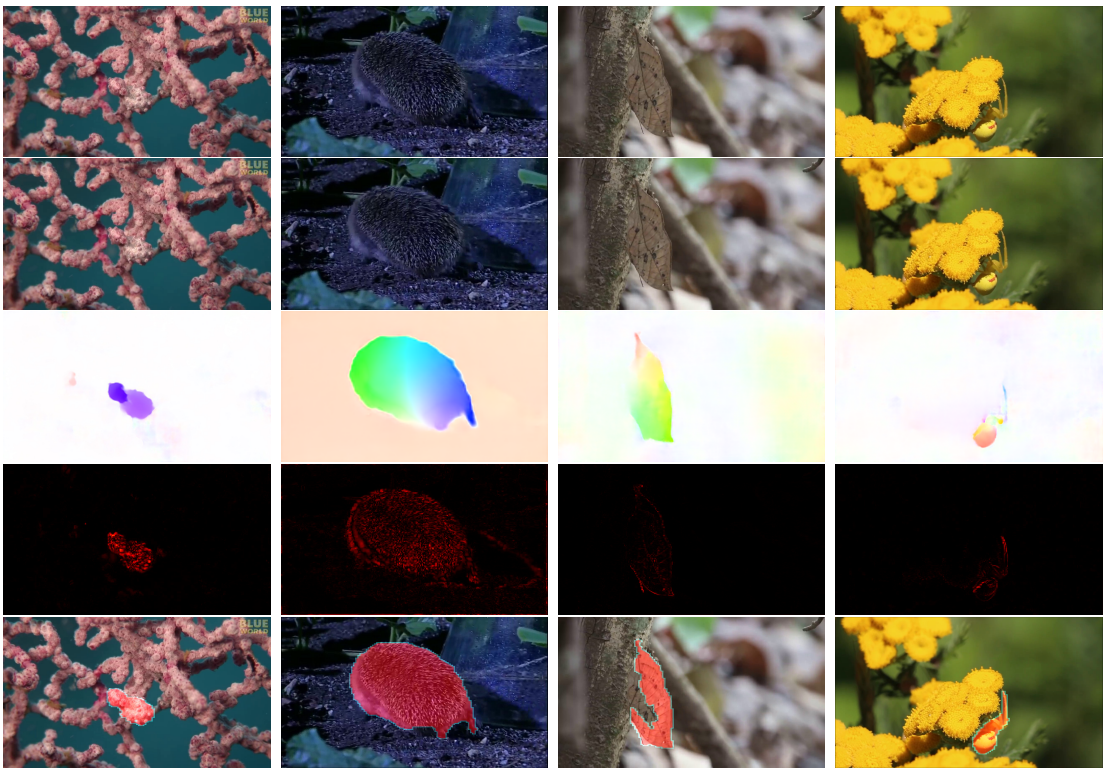


Figure 2.6: Motion segmentation results on MoCA. From top to bottom: frame t, frame $t+1$, PWCNet optical flow, aligned image difference, moving object segmentation.

As shown in Table 2.3, when compared with MPNet which also take flow-only input, our model (ours-C) outperforms it on all metrics by a large margin. Note that, the DAVIS benchmark is fundamentally different from MoCA, as the approaches only relying on appearance are very effective [Z. Yang et al. 2019], indicating that the objects in the DAVIS sequences can indeed be well-identified by the appearance, and motion is not playing the dominant role as it is in MoCA. This can also be observed from the results for Seg-det, Seg-track, AnchorDiff and COSNet, which show significantly stronger performance on DAVIS than on MoCA. Given this difference, our flow-based architecture still shows very competitive performance. Moreover, when we extend our model with an RGB appearance stream, we do observe a performance boost, but since our appearance model has only been fine-tuned on the DAVIS training set (30 training sequences), the two stream (ours-E) is not comparable with other models trained with more segmentation data.

		Flow-based		RGB-based		Two-stream			
	Measure	MPNet	**ours-C**	AD	COSNet	LVO	Seg-det	Seg-track	**ours-E**
$\mathcal{J}$	Mean	60.3	**65.3**	**80.3**	77.6	69.8	**76.8**	75.8	69.9
	Recall	69.9	**77.3**	90.0	**91.4**	83.8	84.8	81.8	**85.3**
$\mathcal{F}$	Mean	58.7	**65.1**	**79.3**	77.5	70.1	**77.8**	76.1	70.3
	Recall	64.3	**74.7**	84.7	**87.4**	84.3	**91.3**	88.7	82.9

Table 2.3: Results on the validation set of DAVIS 2016.

2.7 Conclusions

To summarise, in this paper we consider the problem of breaking animal camouflage in videos. Specifically, we propose a novel and effective architecture with two components: a differentiable registration module to highlight object boundaries; and a motion segmentation module with memory that discovers moving regions. As future work, we propose to improve the architecture from two aspects: *First*, building more effective memory module for handling longer video sequences, for example, a Transformer [Vaswani et al. 2017]. *Second*, for objects that are only partially moving, RGB appearance is required to get the sense of objectness, therefore a future direction is to explore RGB inputs, for both visual-matching-based registration and appearance features.

Acknowledgements

This research was supported by the UK EPSRC CDT in AIMS, Schlumberger Studentship, and the UK EPSRC Programme Grant Seebibyte EP/M013774/1.

Appendices

The appendices can be found on the online version of the paper.

Statement of authorship

A statement of authorship for this paper is provided in the Appendix.

Chapter 3

Self-supervised video object segmentation by motion grouping

The paper was published at the International Conference on Computer Vision, 2021. A short version of this paper also won the Best Paper Award at CVPR Workshop on Robest Video Scene Understanding, 2021.

turbulent times (with COVID-19)

Generated with Meta AI

Self-supervised video object segmentation by motion grouping

Charig Yang Hala Lamdouar Erika Lu

Andrew Zisserman Weidi Xie

VGG, Department of Engineering Science, University of Oxford

{charig,lamdouar,erika,az,weidi}@robots.ox.ac.uk

https://www.charigyang.github.io/motiongroup

Abstract

Animals have evolved highly functional visual systems to understand motion, assisting perception even under complex environments. In this paper, we work towards developing a computer vision system able to segment objects by exploiting motion cues, i.e. motion segmentation. To achieve this, we introduce a simple variant of the Transformer to segment optical flow frames into primary objects and the background, which can be trained in a self-supervised manner, i.e. without using any manual annotations. Despite using only optical flow as input, our approach achieves superior results compared to previous state-of-the-art self-supervised methods on public benchmarks (DAVIS2016, SegTrackv2, and FBMS59), while being an order of magnitude faster. We additionally evaluate on a challenging camouflage dataset (MoCA), significantly outperforming the other self-supervised approaches, and competitive to the top supervised approach, highlighting the importance of motion cues, and the potential bias towards appearance in existing video segmentation models.

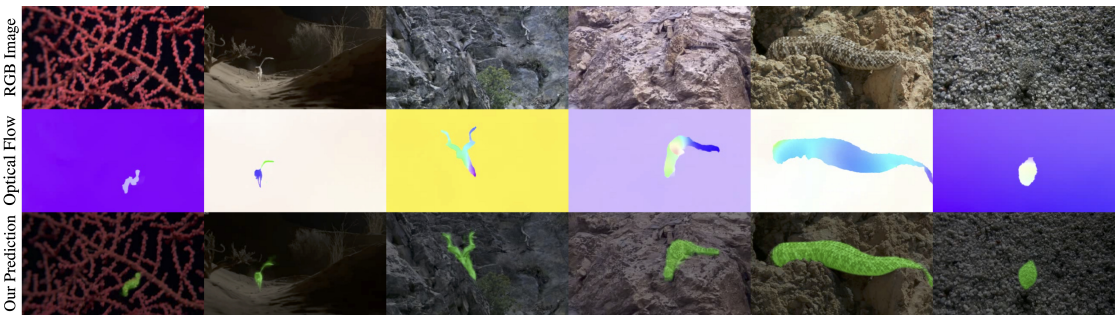


Figure 3.1: **Segmenting camouflaged animals.** Motion plays a critical role in augmenting the capability of our visual system for perceptual grouping in complex scenes – for example, in these sequences (MoCA dataset [Lamdouar et al. 2020]), the visual appearance (RGB images) is clearly uninformative. In this paper, we propose a self-supervised approach to segment objects using *only* motion, i.e. optical flow. From top to bottom rows, we show the video frames, optical flow between consecutive frames, and the segmentation produced by our approach.

3.1 Introduction

When looking around the world, we effortlessly perceive a complex scene as a set of distinct objects. This phenomenon is referred to as *perceptual grouping* – the process of organizing the incoming visual information – and is usually considered a fundamental cognitive ability that enables understanding and interacting with the world efficiently. How do we accomplish such a remarkable perceptual achievement, given that the visual input is, in a sense, just a spatial distribution of variously colored individual points/pixels? In 1923, Wertheimer [Wertheimer 1923] first introduced the *Gestalt principles* with the goal of formulating the underlying causes by which sensory data is organized into groups, or Gestalten. The principles are much like heuristics with "a bag of tricks" [Ramachandran 1985] that the visual system may exploit for grouping, for example, proximity, similarity, closure, continuation, common fate, *etc.*

In computer vision, *perceptual grouping* is often closely related to the problem of segmentation, i.e. extracting the objects with arbitrary shape (pixel-wise labels) from cluttered scenes. In the recent literature of semantic or instance segmentation, tremendous progress has been made by training deep neural networks on image or video datasets. While it is exciting to see machines with the ability to detect, segment, and classify objects in images or video frames, training such segmentation models through supervised learning requires massive human annotation, and consequently limiting their scalability. Even more importantly, the

assumption that objects can be well-identified by their appearance alone in static frames is often an oversimplification – objects are not always visually distinguishable from their background environment. For instance when trying to discover camouflaged animals/objects from the background (Figure 3.1), extra cues, such as motion or sound, are usually required.

Among the numerous cues, motion is usually simple to obtain as it can be implicitly generated from unlabeled videos. In this paper, we aim to exploit such cues for object segmentation in a self-supervised manner, i.e. *zero* human annotation is required for training. At a high level, we aim to exploit the common fate principle, with the basic assumption being that **elements tend to be perceived as a group if they move in the same direction at the same rate (have similar optical flow)**. Specifically, we tackle the problem by training a generative model that decomposes the optical flow into foreground (object) and background layers, describing each as a homogeneous field, with discontinuities occurring only between layers. We adopt a variant of the Transformer [Vaswani et al. 2017], with the self-attention being replaced by slot attention [Locatello et al. 2020], where iterative grouping and binding have been built into the architecture. With some critical architectural changes, we show that pixels undergoing similar motion are grouped together and assigned to the same layer.

To summarize, we make the following contributions: *first*, we introduce a simple architecture for video object segmentation by exploiting motions, using only optical flow as input. *Second*, we propose a self-supervised proxy task that is used to train the architecture without any manual supervision. To validate these contributions, we conduct thorough ablation studies on the components that are key to the success of our architecture, such as a consistency loss on optical flow computed from various frame gaps. We evaluate the proposed architecture on public benchmarks (DAVIS2016 [Perazzi et al. 2016], SegTrackv2 [F. Li et al. 2013], and FBMS59 [Ochs et al. 2014]), outperforming previous state-of-the-art self-supervised models, with comparable performance to the supervised approaches. Moreover, we also evaluate on a camouflage dataset (MoCA [Lamdouar et al. 2020]), demonstrating a significant performance improvement over the other self- and supervised approaches, highlighting the importance of motion cues, and the potential bias towards visual appearance in existing video segmentation models.

3.2 Related work

Video object segmentation has been a longstanding task in computer vision, which involves assigning pixels (or edges) of an image into groups (e.g. objects). In recent literature [Bideau and Learned-Miller 2016a; Pont-Tuset et al. 2017; Brox and Malik 2010; Ochs and Brox 2011; Papazoglou and Ferrari 2013; Jain et al. 2017; Tokmakov et al. 2019; Dave et al. 2019; S. W. Oh et al. 2019; Voigtlaender et al. 2019; Vondrick et al. 2018; W. Wang et al. 2019; Lai and W. Xie 2019; Lai et al. 2020; Maninis et al. 2018; Voigtlaender and Leibe 2017; Caelles et al. 2017; Fragkiadaki et al. 2012; Keuper et al. 2015; Jun Koh and C.-S. Kim 2017; Fan et al. 2019; Z. Yang et al. 2019; Pont-Tuset et al. 2017; Xu et al. 2018; L. Liu et al. 2020], two protocols have attracted increasing interest from the vision community, namely, semi-supervised video object segmentation (**semi-supervised VOS**), and unsupervised video object segmentation (**unsupervised VOS**). The former aims to re-localize one or multiple targets that are specified in the first frame of a video with pixel-wise masks, and the latter considers automatically segmenting the object of interest (usually the most salient one) from the background in a video sequence. Despite being called **unsupervised VOS**, in practice, the popular methods to address such problems extensively rely on supervised training, for example, by using two-stream networks [Papazoglou and Ferrari 2013; Jain et al. 2017; Tokmakov et al. 2019; Dave et al. 2019] trained on large-scale external datasets. As an alternative, in this work, we consider a *completely unsupervised* approach, where no manual annotation is used for training whatsoever.

Motion segmentation shares some similarity with unsupervised VOS, but focuses on discovering *moving* objects. In the literature, [Brox and Malik 2010; Ochs and Brox 2011; Keuper et al. 2015; Sivic et al. 2004; C. Xie et al. 2019] consider clustering the pixels with similar motion patterns; [Tokmakov et al. 2017; Tokmakov et al. 2019; Dave et al. 2019] train deep networks to map the motions to segmentation masks. Another line of work has tackled the problem by explicitly leveraging the independence of motion between the moving object and its background. For instance, [Y. Yang et al. 2019] proposes an adversarial setting, where a generator is trained to produce masks, altering the input flow, such that the inpainter fails to estimate the missing information. In [Bideau and Learned-Miller

2016b; Bideau et al. 2018; Lamdouar et al. 2020], the authors propose to highlight the independently moving object by compensating for the background motion, either by registering consecutive frames, or explicitly estimating camera motion. In constrained scenarios, such as autonomous driving, [Ranjan et al. 2019] proposes to jointly optimize depth, camera motion, optical flow and motion segmentation.

Optical flow computation is one of the fundamental tasks in computer vision. Deep learning methods allow efficient computation of optical flow, both in training on synthetic data [D. Sun et al. 2018; Teed and Deng 2020], or learning with photometric loss in self-supervised [L. Liu et al. 2020; P. Liu et al. 2019] setting. In practise, flow has been useful for a wide range of problems, for example, pose estimation [Doersch and Zisserman 2019], representation learning [Mahendran et al. 2018a; T. Han et al. 2020], segmentation [Brox and Malik 2010], and occasionally even used in lieu of appearance cues (RGB images) for tracking [Sidenbladh et al. 2000].

Transformer architectures have proven extremely adept at modelling long-term relationships within an input sequence via attention mechanisms. Originally used for language tasks [Vaswani et al. 2017; Devlin et al. 2018; T. B. Brown et al. 2020], they have since been adapted to solve popular computer vision problems such as image classification [Dosovitskiy et al. 2021], generation [M. Chen et al. 2020; Ramesh et al. 2021], video understanding [X. Wang et al. 2018; Girdhar et al. 2019; Bertasius et al. 2021], object detection [Carion et al. 2020], and zero-shot classification [Radford et al. 2021]. In this work, we take inspiration from a specific variant of self-attention, namely slot attention [Locatello et al. 2020], which was demonstrated to be effective for learning object-centric representations on synthetic data, e.g. CLEVR [Johnson et al. 2017].

Layered representations were originally proposed by Wang and Adelson [J. Y. Wang and Adelson 1994] to represent a video as a composition of layers with simpler motions. Since then, layered representations have been widely adopted in computer vision [Brostow and Essa 1999; Jojic and Frey 2001; Zitnick et al. 2004; Kumar et al. 2008; Xue et al. 2015], often to estimate optical flow [D. Sun et al.

2012; D. Sun et al. 2013; Wulff and Black 2014; Wulff and Black 2015]. More recently, deep learning-based layer decomposition methods have been used to infer depth for novel view synthesis [T. Zhou et al. 2018; Srinivasan et al. 2019], separate reflections and other semi-transparent effects [Alayrac et al. 2019b; Alayrac et al. 2019a; Gandelsman et al. 2019; E. Lu et al. 2020], or perform foreground/background estimation [Gandelsman et al. 2019]. These works operate on RGB inputs and produce RGB layers, whereas we propose a layered decomposition of optical flow inputs for unsupervised moving object discovery.

Object-centric representations interpret scenes with "objects" as the basic building blocks (instead of individual pixels), which is considered an essential step towards human-level generalization. There is a rich literature on this topic, for example, IODINE [Greff et al. 2019] uses iterative variational inference to infer a set of latent variables recurrently, with each representing one object in an image. Similarly, MONet [Burgess et al. 2019] and GENESIS [Engelcke et al. 2020] also adopt multiple encoding-decoding steps. In contrast, [Locatello et al. 2020] proposes Slot Attention, which enables single step encoding-decoding with iterative attention. However, all works mentioned above have only shown applications for synthetic datasets, e.g. CLEVR [Johnson et al. 2017]. In this paper, we are the first to demonstrate its use for object segmentation of realistic videos by exploiting motion, where the challenging nuances in visual appearance (e.g. the complex background textures) have been removed.

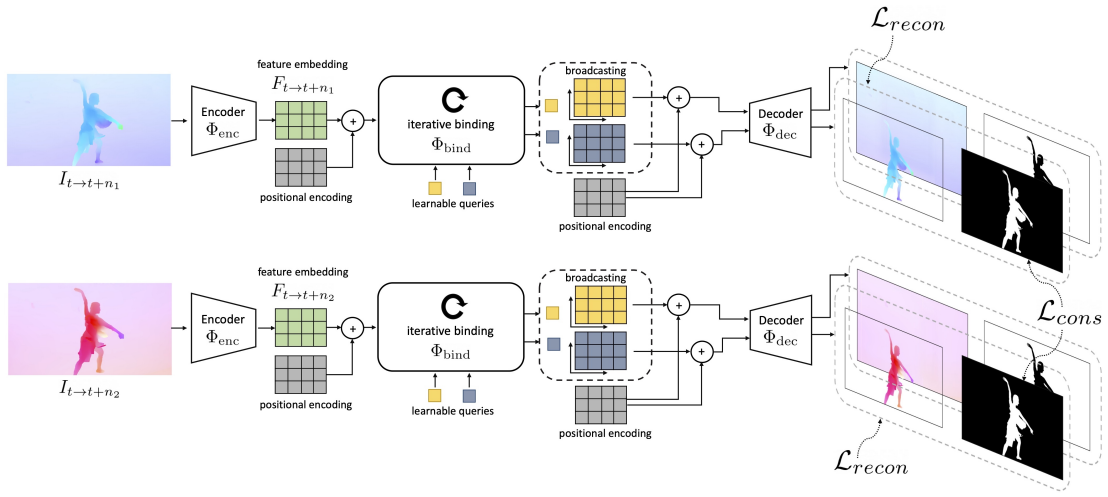


Figure 3.2: **Pipeline.** Our model takes optical flow $I_{t\to t+n}$ as input, and outputs a set of reconstruction and opacity layers. Specifically, it consists of three components: feature encoding, iterative binding, and decoding to layers, which are combined to reconstruct the input flow. To resolve motion ambiguities (small motion), or noise in optical flow, consistency between two flow fields computed under different frame gaps is enforced during training. At inference time, only the top half of the figure is used to predict masks from a single-step flow.

3.3 Method

Our goal is to take an input optical flow frame and predict a segment containing the moving object. We propose to train this model in a self-supervised manner, with an autoencoder-type framework. Specifically, our model outputs two layers: one representing the background, and the other for one or more moving objects in the foreground, as well as their oppacity layers (weighted masks). Formally:

$$\{\hat{I}^i_{t\to t+n}, \alpha^i_{t\to t+n}\}_{i=1}^N = \Phi(I_{t\to t+n}) \tag{3.1}$$

where $I_{t\to t+n}$ refers to the t to $t+n$ input flow (backward flow when $n<0$), $\Phi(\cdot)$ is the parametrized model, $\hat{I}^i_{t\to t+n}$ is the ith layer reconstruction, $\alpha^i_{t\to t+n}$ is its mask, and $N=2$ is the number of layers (foreground and background). These layers can then be composited linearly to reconstruct the input image $I_{t\to t+n}$:

$$\hat{I}_{t\to t+n} = \sum_{i=1}^{N} \alpha^i_{t\to t+n} \hat{I}^i_{t\to t+n} \tag{3.2}$$

3.3.1 Flow segmentation architecture

For simplicity, we first consider the case of a single flow field as input (depicted in the top part of Figure 3.2). The entire model consists of three components: (1) a CNN encoder to extract a compact feature representation, (2) an iterative binding module with learnable queries that plays a similar role as soft clustering, i.e. assigning each pixel to one of motion groups, and (3) a CNN decoder that individually decodes each query to full resolution layer outputs (where thresholding the alpha channel yields the predicted segment).

CNN encoder. We first pass the precomputed optical flow between two frames, $I_{t\to t+n} \in \mathcal{R}^{3\times H_0\times W_0}$, to a CNN encoder Φ_{enc}, which outputs a lower-resolution feature map:

$$F_{t\to t+n} = \Phi_{\text{enc}}(I_{t\to t+n}) \in \mathcal{R}^{D\times H\times W} \tag{3.3}$$

where H_0, W_0 and H, W refer to the spatial dimensions of the input and output feature maps respectively. Note that, we convert the flow into a three-channel image using the traditional method in the optical flow literature [D. Sun et al. 2018].

Iterative binding. The iterative binding module Φ_{bind} aims to group image regions into single entities based on their similarities in motion, i.e. pixels moving in the same direction at the same rate should be grouped together. Intuitively, such a binding process requires a data-dependent parameter updating mechanism, iteratively enriching the model, gradually including more pixels undergoing similar motions.

To accomplish this task, we adopt a simple variant of slot attention [Locatello et al. 2020], where instead of Gaussian-initialized slots, we use learnable query vectors. Slot attention has recently shown remarkable performance for object-centric representation learning, where the query vectors compete to explain parts of the inputs via a softmax-based attention mechanism, and the representations of these slots are iteratively updated with a recurrent update function. In our case of motion segmentation, ideally, the final representation in each query vector separately

encodes the moving object or the background, which can then be decoded and combined to reconstruct the input flow fields.

Formally, our inputs to Φ_{bind} are feature maps $F_{t\to t+n}$ and two learnable queries (representing foreground and background) $Q \in \mathcal{R}^{D\times 2}$. Learnable spatial positional encodings are summed with $F_{t\to t+n}$; with some abuse of notation, we still refer to this sum as $F_{t\to t+n}$. We use *three* different linear transformations to generate the *query* , *key* and *value*: $q \in \mathcal{R}^{D\times 2}$, $k, v \in \mathcal{R}^{D\times HW}$,

$$q,\ k,\ v = W^Q{\cdot}Q,\ W^K{\cdot}F_{t\to t+n},\ W^V{\cdot}F_{t\to t+n} \tag{3.4}$$

where $W^Q, W^K, W^V \in \mathcal{R}^{D\times D}$.

In contrast to the standard Transformer [Vaswani et al. 2017], the coefficients in slot attention are normalized over all slots. This choice of normalization introduces competition between the slots to explain parts of the input, and ensures each pixel is assigned to a query vector:

$$\begin{aligned} \text{attn}_{i,j} &:= \frac{e^{M_{i,j}}}{\sum_l e^{M_{i,l}}} \\ M &:= \frac{1}{\sqrt{D}} k^T{\cdot}q,\ \ \text{attn} \in \mathcal{R}^{HW\times 2} \end{aligned} \tag{3.5}$$

To aggregate the input values to their assigned query slot, a weighted mean is used as follows:

$$\begin{aligned} U &:= v{\cdot}A \ \ \in \mathcal{R}^{D\times 2} \\ \text{where,}\ \ A_{i,j} &:= \frac{\text{attn}_{i,j}}{\sum_l \text{attn}_{l,j}} \end{aligned} \tag{3.6}$$

To maintain a smooth update of the query slots Q, the aggregated vectors U are fed into a recurrent function, parametrized with Gated Recurrent Units (GRU),

$$Q := \text{GRU}(\text{inputs} = U, \text{states} = Q) \tag{3.7}$$

This whole binding process is then iterated T times. The pseudocode can be found in the Supplementary Material.

CNN decoder. The CNN decoder Φ_{dec} individually decodes each of the slots to outputs of original resolution ($\{\hat{I}^i_{t\to t+n}, \alpha^i_{t\to t+n}\} \in \mathcal{R}^{4\times H_0\times W_0}$), which includes an (unnormalized) single-channel alpha mask and the reconstructed flow fields. Specifically, the input to the decoder is the slot vector broadcasted onto a 2D grid augmented with a learnable spatial positional encoding.

Reconstruction. Once each slot has been decoded, we apply softmax to the alpha masks across the slot dimension, and use them as mixture weights to obtain the reconstruction $\hat{I}_{t\to t+n}$ (Eq. 3.2). Our reconstruction loss is an L2 loss between the input and reconstructed flow,

$$\mathcal{L}_{recon} = \frac{1}{\Omega}\sum_{p\in\Omega} |I_{t\to t+n}(p) - \hat{I}_{t\to t+n}(p)|^2 \tag{3.8}$$

where p is the pixel index, and Ω is the entire spatial grid.

Entropy regularization. We impose a pixel-wise entropy regularisation on inferred masks:

$$\mathcal{L}_{entr} = \frac{1}{\Omega}\sum_{p\in\Omega}(-\alpha^1_{t\to t+n}(p)\log\alpha^1_{t\to t+n}(p) - \alpha^0_{t\to t+n}(p)\log\alpha^0_{t\to t+n}(p)) \tag{3.9}$$

This loss is zero when the alpha channels are one-hot, and maximum when they are of equal probability. Intuitively, this helps encourage the masks to be binary, which aligns with our goal in obtaining segmentation masks.

Instance normalisation. In the case of motion segmentation, objects can only be detected if they undergo an independent motion from the camera; thus previous work attempts to compensate for camera motion [Bideau and Learned-Miller 2016b; Lamdouar et al. 2020]. We are inspired by these ideas, but instead of explicitly estimating homography or camera motion, we take a poor-man's approach by simply using Instance Normalisation (IN) [Ulyanov et al. 2016] in the CNN encoder and decoder, which normalizes each channel of the training sample independently. Intuitively, the *mean* activation tends to be dominated by the motions in the large homogeneous region, which is usually the background. This

normalization, in combination with ReLU activations, helps gradually separate the background motion from the foreground motions. This is experimentally shown in Section 3.5.1.

3.3.2 Self-supervised temporal consistency loss

The segmentation computed for the current frame should be identical irrespective of whether the 'second' frame is consecutive, or earlier or later in time. We harness this constraint to form a self-supervised temporal consistency loss by first defining a set of 'second' frames, and then requiring consistency between their pairwise predictions. We describe the set first, followed by the loss.

Multi-step flow. As objects may be static for some frames, we make our predictions more robust by leveraging observations from multiple timesteps. We consider the flow fields computed from various temporal gaps as an input set, i.e. $\{I_{t\to t+n_1}, I_{t\to t+n_2}\}$, $n_1, n_2 \in \{-2, -1, 1, 2\}$, and use a permutation invariant consistency loss to encourage the model to predict the same foreground/background segmentation for all flow fields in the set.

Consistency loss. We randomly sample two flow fields from the input set and pass them through the model ($\Phi(\cdot)$), outputting the flow reconstruction and alpha masks for each. As the reconstruction loss is commutative, it is not guaranteed that the same slot will always output the background layer; therefore, we use a permutation-invariant consistency loss, i.e. only backpropagating through the lowest-error permutation:

$$\mathcal{L}_{cons} = \frac{1}{\Omega}\min\left(\sum_{p\in\Omega} |\alpha^1_{t\to t+n_1}(p) - \alpha^1_{t\to t+n_2}(p)|^2, \sum_{p\in\Omega} |\alpha^1_{t\to t+n_1}(p) - \alpha^0_{t\to t+n_2}(p)|^2\right)$$

Note that, this consistency enforcement only occurs during training. At inference time, a single-step flow is used, as shown in the top half of Figure 3.2.

Total loss. The total loss for training the architecture is:

$$\mathcal{L}_{total} = \gamma_r \mathcal{L}_{recon} + \gamma_c \mathcal{L}_{cons} + \gamma_e \mathcal{L}_{entr} \tag{3.10}$$

we use $\gamma_r = 10^2$, $\gamma_c = 10^{-2}$ and $\gamma_e = 10^{-2}$, but we found the model to be fairly robust to these hyperparameters.

3.3.3 Discussion

Differences from slot attention. Slot attention was originally introduced for self-supervised object segmentation for RGB images [Locatello et al. 2020], and its usefulness was demonstrated on synthetic data (CLEVR [Johnson et al. 2017]), where objects are made of primitive shapes with simple textures. However, this assumption is unlikely to hold in the case of natural images or videos, making it challenging to generalize such object-centric representations.

In this work, we build on the insight that although objects in images may not be naturally textureless, their motions typically are. Hence, we develop the self-supervised object segmentation model by exploiting their optical flows, where the nuance in visual appearance is discarded, thus not restricted to simple synthetic cases. As an initial trial, we experimented with the same setting as [Locatello et al. 2020], where query vectors are sampled from a Gaussian distribution; however, we were unable to train it. Instead, we use learnable embeddings here, which we highlight as one of the architectural changes critical to our model's success. Other critical changes include instance normalization and temporal consistency, which we demonstrate in ablations in Section 3.5.1.

Why does it work for motion segmentation? The proposed idea can be seen as training a generative model to segment the flow fields. With the layered formulation, reconstruction is limited to be a simple *linear* composition of layer-wise flow, decoded from a single slot vector.

Conceptually, this design has effectively introduced a representational bottleneck, encouraging each slot vector to represent minimal information, i.e. homogeneous motion, and with minimal redundancy (mutual information) between slots. All these properties make such an architecture well-suited to the task of segmenting

objects undergoing independent motions.

3.4 Experimental setup

3.4.1 Datasets

DAVIS2016 [Perazzi et al. 2016] contains a total of 50 sequences (30 for training and 20 for validation), depicting diverse moving objects such as animals, people, and cars. The dataset contains 3455 1080p frames with pixel-wise annotations at 480p for the predominantly moving object.

SegTrackv2 [F. Li et al. 2013] contains 14 sequences and 976 annotated frames. Each sequence contains 1-6 moving objects, and presents challenges including motion blur, appearance change, complex deformation, occlusion, slow motion, and interacting objects.

FBMS59 [Ochs et al. 2014] consists of 59 sequences and 720 annotated frames (every 20th frame is annotated), which vary greatly in image resolution. Sequences involve multiple moving objects, some of which may be static for periods of time.

Moving Camouflaged Animals (MoCA) [Lamdouar et al. 2020] contains 141 HD video sequences, depicting 67 kinds of camouflaged animals moving in natural scenes. Both temporal and spatial annotations are provided in the form of tight bounding boxes for every 5th frame. Using the provided motion labels (locomotion, deformation, static), we filter out videos with predominantly no locomotion, resulting in 88 video sequences and 4803 frames.

3.4.2 Evaluation metrics

Segmentation (Jaccard). For DAVIS2016, SegTrackv2 and FBMS59, pixel-wise segmentation is provided; thus we report the standard metric, region similarity ($\mathcal{J}$), computing the mean over the test set. For FBMS59 and SegTrackv2, we follow the common practice [Y. Yang et al. 2019; Jain et al. 2017] and combine multiple objects as one single foreground.

Localization (Jaccard & success rate). As the MoCA dataset provides only bounding box annotations, we evaluate for the detection task and report results in the form of detection success rate [Everingham et al. 2010; T.-Y. Lin et al. 2014], for varying IoU thresholds ($\tau \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$).

3.4.3 Implementation details

We evaluate three different approaches for computing optical flow, namely, PWC-Net [D. Sun et al. 2018], RAFT [Teed and Deng 2020] and ARFlow [L. Liu et al. 2020]; the first two are supervised, while the latter is self-supervised. We extract the optical flow at the original resolution of the image pairs, with the frame gaps $n \in \{-2, -1, 1, 2\}$ for all datasets, except for FBMS59, where we use $n \in \{-6, -3, 3, 6\}$ to compensate for small motion. To generate inputs to the network for training, the flows are resized to 128×224 (and scaled accordingly), converted to 3-channel images with the standard visualization used for optical flow, and normalized to $[-1, 1]$.

In the iterative binding module ($\Phi_{\text{bind}}(\cdot)$), we use two learnable query vectors (as we consider the case of segmenting a single moving object from the background), and choose $T = 5$ iterations (as explained in Section 3.3.1). We adopt a simple VGG-style network for the CNN encoder and decoder with instance normalization. We train with a batch size of 64 images and use the Adam optimizer [Kingma and Ba 2015] with an initial learning rate of 5×10^{-4}, decreasing every 80k iterations. The exact architecture description and training schedule can be found in the Supplementary Material.

3.5 Results

In this section, we compare primarily with a top-performing approach trained without manual annotations – Contextual Information Separation (CIS [Y. Yang et al. 2019]). However, as the architecture, input resolution, modality and post-processing are all different, we try our best to conduct the comparison as fairly as possible. Note that benchmarks are evaluated at full resolution by simply upsampling the predicted masks.

Model	Flow	IN	T	$\mathcal{L}_e$	$\mathcal{L}_c$	DAVIS ($\mathcal{J}\uparrow$)
CIS [Y. Yang et al. 2019]	PWC-Net	–	–	–	–	59.2
Ours-A	PWC-Net	✓	5	✓	✓	63.7
Ours-B	**RAFT**	✓	**5**	✓	✓	**68.3**
Ours-C	ARFlow	✓	5	✓	✓	53.2
Ours-D	RAFT	✓	3	✓	✓	65.8
Ours-E	RAFT	✗	3	✓	✓	63.3
Ours-F	RAFT	✗	5	✓	✓	64.5
Ours-G	RAFT	✓	5	✗	✗	48.0
Ours-H	RAFT	✓	5	✗	✓	60.3
Ours-I	RAFT	✓	5	✓	✗	51.2

Figure 3.3: **Ablation studies** on flow extraction methods, instance normalization (IN), grouping iterations (T), entropy regularization ($\mathcal{L}_e$) and set consistency ($\mathcal{L}_c$).

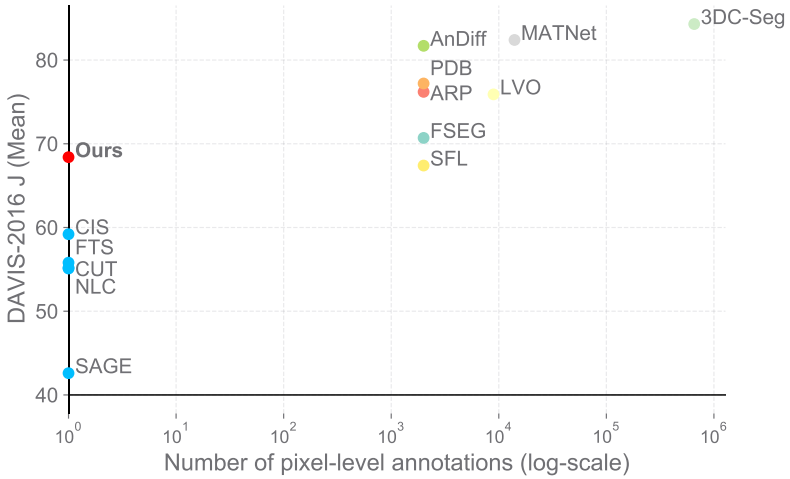


Figure 3.4: **Comparison on DAVIS2016.** Note that, supervised approaches may use ImageNet pretraining [Deng et al. 2009], but here we only count images with pixel-wise annotations.

Model	Sup.	RGB	Flow	Res.	D16	STv2	FBMS	Runtime
SAGE [W. Wang et al. 2018]	✗	✓	✓	–	42.6	57.6	**61.2**	0.9s
NLC [Faktor and Irani 2014]	✗	✓	✓	–	55.1	**67.2**	51.5	11s
CUT [Keuper et al. 2015]	✗	✓	✓	–	55.2	54.3	57.2	103s
FTS [Papazoglou and Ferrari 2013]	✗	✓	✓	–	55.8	47.8	47.7	0.5s
CIS [Y. Yang et al. 2019]	✗	✓	✓	192 × 384	59.2 (71.5)	45.6 (62.0)	36.8 (63.5)	0.1s (11s)
Ours	✗	✗	✓	128 × 224	**68.3**	58.6	53.1	**0.012s**
SFL [J. Cheng et al. 2017]	✓	✓	✓	854 × 480	67.4	–	–	7.9s
FSEG [Jain et al. 2017]	✓	✓	✓	854 × 480	70.7	61.4	68.4	–
LVO [Tokmakov et al. 2019]	✓	✓	✓	–	75.9	57.3	65.1	–
ARP [Song et al. 2018]	✓	✓	✓	–	76.2	57.2	59.8	74.5s
COSNet [X. Lu et al. 2019]	✓	✓	✗	473 × 473	80.5	–	75.6	–
MATNet [T. Zhou et al. 2020]	✓	✓	✓	473 × 473	82.4	–	–	0.55s
3DC-Seg [Mahadevan et al. 2020]	✓	✓	✓	854 × 480	84.3	–	–	0.84s

Table 3.1: **Full comparison on moving object segmentation** (unsupervised video segmentation). We consider three popular datasets, DAVIS2016, SegTrack-v2 (STv2), and FBMS59. Models above the horizontal dividing line are trained without using *any* manual annotation, while models below require ground truth annotations at training time. Numbers in parentheses denote the additional usage of significant post-processing, e.g. multi-step flow, multi-crop, temporal smoothing, CRFs. Runtime excludes optical flow computation.

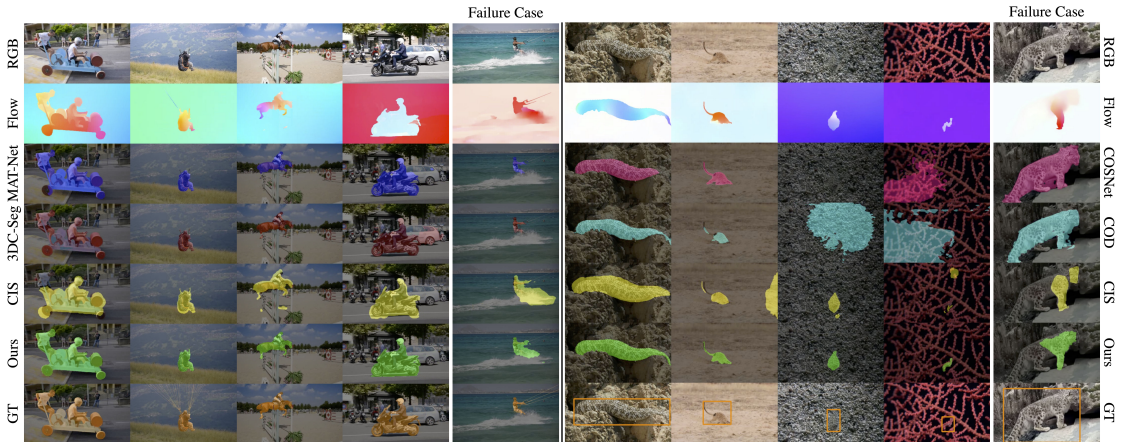


Figure 3.5: **Qualitative results.** On DAVIS2016 (left), our method is able to segment a variety of challenging objects, often on-par with top supervised approaches. On MoCA (right), our model is able to accurately segment well-camouflaged objects even when previous supervised methods fail completely (3rd, 4th columns). We show a failure case (left) where the splash created by the person is incorrectly included in our predicted segment, and another failure case (right) where the animal is only partially moving and thus partially segmented.

					Success Rate					
Model	Sup.	RGB	Flow	$\mathcal{J}\uparrow$	$\tau = 0.5$	$\tau = 0.6$	$\tau = 0.7$	$\tau = 0.8$	$\tau = 0.9$	SR_{mean}
COD [Lamdouar et al. 2020]	✓	✗	✓	44.9	0.414	0.330	0.235	0.140	0.059	0.236
COD (two-stream)	✓	✓	✓	55.3	0.602	0.523	0.413	0.267	0.088	0.379
COSNet [X. Lu et al. 2019]	✓	✓	✗	50.7	0.588	0.534	0.457	0.337	0.167	0.417
MATNet [T. Zhou et al. 2020]	✓	✓	✓	**64.2**	**0.712**	**0.670**	**0.599**	**0.492**	**0.246**	**0.544**
CIS	✗	✓	✓	49.4	0.556	0.463	0.329	0.176	0.030	0.311
CIS (post-processing)	✗	✓	✓	54.1	0.631	0.542	0.399	0.210	0.033	0.363
Ours	✗	✗	✓	**63.4**	**0.742**	**0.654**	**0.524**	**0.351**	**0.147**	**0.484**

Table 3.2: **Comparison results on MoCA dataset.** We report the successful localization rate for various thresholds τ (see Section 3.4.2). Both CIS and Ours were pre-trained on DAVIS and finetuned on MoCA in a self-supervised manner. Our method achieves comparable Jaccard ($\mathcal{J}$) to MATNet (2nd best model on DAVIS), without using RGB inputs and without any manual annotation for training.

3.5.1 Ablation studies

We conduct all ablation studies on DAVIS2016, and vary one variable each time, as shown in Table 3.3.

Choice of optical flow algorithm. With the same flow extraction method (PWC-Net), our proposed model (Ours-A) outperforms CIS by about 4.5 points on mean Jaccard ($\mathcal{J}$), and using improved optical flow (RAFT) provides further performance gains. We therefore use RAFT from hereon.

Instance normalization and grouping. We observe two phenomena: *first*, when holding constant on the number of grouping iterations T (3 or 5), models trained with instance normalization perform consistently better; *second*, iterative grouping with $T = 5$ is better than that trained with $T = 3$. However, at $T = 8$, the model did not converge in the same number of training steps, and thus we do not include it in the table. For the remainder of the experiments, we use instance normalization and $T = 5$.

Consistency and entropy regularization. While comparing Ours-B and Ours-I, we observe that the performance degrades significantly without the temporal consistency loss, and that the entropy regularization is also important, as shown by Ours-B and Ours-H.

3.5.2 Comparison with state-of-the-art

We show our results in Table 3.1. On DAVIS2016, we improve upon the state-of-the-art for unsupervised methods (CIS) by a large margin (+9.1%). As shown in Figure 3.4, despite not using any pixel-level annotations during training, our method is nearing the performance of supervised models trained on thousands of images.

In addition, we argue that, motion segmentation in realistic scenarios, e.g. by predator or prey, is likely to require fast processing. Our model operates on small resolution (potentially sacrificing some accuracy) with over 80fps. Our method's efficiency gain mainly comes from two sources: first, our model is a lightweight VGG-style network with only 4.77**M** parameters; second, we disregard any post-processing used in previous approaches, e.g. averaging the prediction across multiple flow steps, across multiple crops, temporal smoothing, or CRFs, which cost over 10s in total.

For SegTrackv2 and FBMS59, they occasionally include multiple objects in a single video, and only a subset of them are moving, making it challenging to spot all objects using flow-only input, but we achieve competitive performance nonetheless. We discuss this limitation below.

3.5.3 Camouflage breaking

In addition, we also benchmark the model on camouflaged object detection on MoCA dataset, where visual cues are often less effective than motion cues. To compare fairly with CIS [Y. Yang et al. 2019], we use the code and model released by the authors, and fine-tune their model on MoCA in a self-supervised manner. We convert the output segmentation mask into a bounding box by drawing a bounding box around the largest connected region in the predicted mask.

We report quantitative results in Table 3.2 and show qualitative results in Figure 3.5. Our model significantly outperforms CIS (14% when allowing no post-processing), previous supervised approaches e.g. COD [Lamdouar et al. 2020] (18.5% on Jaccard), and even COSNet [X. Lu et al. 2019] (among the top supervised approaches on DAVIS). We conjecture that COSNet's weaker performance is due to its sole reliance on visual appearance (which is distracting for the MoCA data) rather than using motion inputs. This is particularly interesting, as it clearly indicates that no single information cue is able to do the task perfectly, echoing the two-stream hypothesis [Goodale and Milner 1992] that both appearance and motion are essential to visual systems.

3.5.4 Limitations

Despite showing remarkable improvements on motion segmentation in accuracy and runtime, we note the following limitations of the proposed approach (shown in Figure 3.5) and treat them as future work: *first*, the existing benchmarks are mostly limited to motion segmentation into foreground and background, thus, we choose to use two slots in this paper; however, in real scenarios, videos may contain multiple independently moving objects, which the current model will assign to a single layer. It may be desirable to further separate these objects into different layers. *Second*, we have only explored motion (optical flow) as input, which significantly limits the model in segmenting objects when flow is uninformative or incomplete (as in Figure 3.5, right); however, the self-supervised video object segmentation objective is applicable also to a two-stream approach, and so RGB could be incorporated. *Third*, the current method may fail when optical flow is noisy or low-quality (Figure 3.5, left); jointly optimizing flow and segmentation is a possible way forward in this case.

3.6 Conclusion

In this paper, we present a self-supervised model for motion segmentation. The algorithm takes only flow as input, and is trained without any manual annotation, surpassing previous self-supervised methods on public benchmarks such as DAVIS2016, narrowing the gap with supervised methods. On the more challenging camouflage dataset (MoCA), our model actually compares favourably to the top approaches in video object segmentation that are trained with heavy supervision. As computation power grows and more high quality videos become available, we believe that self-supervised learning algorithms can serve as a strong competitor to the supervised counterparts for their scalability and generalizability.

Acknowledgements

This research is supported by Google-DeepMind Studentship, UK EPSRC CDT in AIMS, Schlumberger Studentship, and the UK EPSRC Programme Grant Visual AI (EP/T028572/1).

Appendices

The appendices can be found on the online version of the paper.

Statement of authorship

A statement of authorship for this paper is provided in the Appendix.

Chapter 4

Moving Object Segmentation: All you need is SAM (and flow)

The paper was published at the Asian Conference on Computer Vision, 2024 as an Oral Presentation.

running out of time (jokes)

Generated with Meta AI

Moving Object Segmentation: All you need is SAM (and flow)

Junyu Xie[1] Charig Yang[1] Weidi Xie[1,2] Andrew Zisserman[1]

[1]VGG, Department of Engineering Science, University of Oxford

[2]Shanghai Jiao Tong University

{jyx,charig,weidi,az}@robots.ox.ac.uk

http://www.robots.ox.ac.uk/~vgg/research/flowsam

Abstract

The objective of this paper is motion segmentation – discovering and segmenting the moving objects in a video. This is a much studied area with numerous careful, and sometimes complex, approaches and training schemes including: self-supervised learning, learning from synthetic datasets, object-centric representations, amodal representations, and many more. Our interest in this paper is to determine if the Segment Anything model (SAM) can contribute to this task.

We investigate two models for combining SAM with optical flow that harness the segmentation power of SAM with the ability of flow to discover and group moving objects. In the first model, we adapt SAM to take optical flow, rather than RGB, as an input. In the second, SAM takes RGB as an input, and flow is used as a segmentation prompt. These surprisingly simple methods, without any further modifications, outperform all previous approaches by a considerable margin in both single and multi-object benchmarks. We also extend these frame-level segmentations to sequence-level segmentations that maintain object identity. Again, this simple model achieves outstanding performance across multiple moving object segmentation benchmarks.

4.1 Introduction

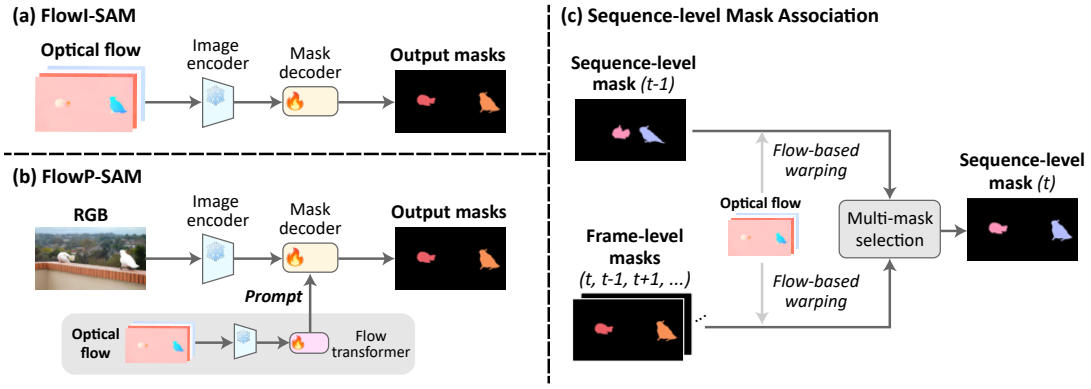


Figure 4.1: **Adapting SAM for Video Object Segmentation by incorporating flow. (a) Flow-as-Input**: `FlowI-SAM` takes in optical flow *only* and predicts frame-level segmentation masks. **(b) Flow-as-Prompt**: `FlowP-SAM` takes in RGB and applies flow information as a prompt for frame-level segmentation. **(c) Sequence-level mask association**: as a post-processing step, the multi-mask selection module autoregressively transforms *frame-level* mask outputs from `FlowI-SAM` and/or `FlowP-SAM` and produces *sequence-level* masks in which object identities are consistent throughout the sequence.

Recent research in image segmentation has been transformative, with the Segment Anything Model (SAM) [Kirillov et al. 2023] emerging as a significant breakthrough. Leveraging large-scale datasets and scalable self-labelling, SAM enables flexible image-level segmentation across many scenarios [J. Ma et al. 2024; Wu et al. 2023; Kirillov et al. 2023; Tang et al. 2023; X. Zhang et al. 2023; T. Chen et al. 2023], facilitated by user prompts such as boxes, texts, and points. In videos, optical flow has played an important and successful role for *moving object segmentation* – in that it can (i) discover moving objects, (ii) provide crisp boundaries for segmentation, and (iii) group parts of objects together if they move together. It has formed the basis for numerous methods of moving object discovery by self-supervised learning [C. Yang et al. 2021; Meunier and Bouthemy 2023; J. Xie et al. 2022; Lamdouar et al. 2021; Meunier et al. 2022; Choudhury et al. 2022; Safadoust and Güney 2023]. However, it faces segmentation challenges if objects are momentarily motionless, and in distinguishing foreground objects from background 'noise'. This naturally raises the question: "How can we leverage the power of SAM with flow for moving object segmentation in videos?".

To this end, we explore two simple variants to effectively tailor SAM for motion segmentation. *First*, we introduce `FlowI-SAM` (Fig. 4.1a), an adaption of the original SAM that directly processes optical flow as a three-channel *input* image for

segmentation, where points on a uniform grid are used as prompts. This approach leverages the ability of SAM to accurately segment moving objects against the static background, by exploiting the distinct textures and clear boundaries present in optical flow fields. However, it has less success in scenes where the optical flow arises from multiple interacting objects as the flow only contains limited information for separating them. *Second*, building on the strong ability of SAM on RGB image segmentation, we propose `FlowP-SAM` (Fig. 4.1b) where the input is an RGB frame, and flow guides SAM for moving object segmentation as *prompts*, produced by a trainable prompt generator. This method effectively leverages the ability of SAM on RGB image segmentation, with flow information acting as a selector of moving objects/regions within a frame. Additionally, we extend these methods from frame-level to *sequence-level* video segmentation (Fig. 4.1c) so that object identities are consistent throughout the sequence. We do this by introducing a matching module that auto-regressively chooses whether to select a new object or propagate the old one based on temporal consistency.

In summary, this paper introduces and explores two models to leverage SAM for moving object segmentation in videos, enabling the principal moving objects to be discriminated from background motions. Our contributions are threefold:

- The `FlowI-SAM` model, which utilises optical flow as a three-channel input image for precise *per-frame* segmentation and moving object identification.
- The `FlowP-SAM` model, a novel combination of dual-stream (RGB and flow) data, that employs optical flow to generate prompts, guiding SAM to identify and localise the moving objects in RGB images.
- New state-of-the-art unsupervised video object segmentation performance by a large margin on moving object segmentation benchmarks, including DAVIS16, DAVIS17-m, YTVOS18-m, and MoCA.

4.2 Related work

Video Object Segmentation (VOS) is an extensively studied task in computer vision. The objective is to segment the primary object(s) in a video sequence. Numerous benchmarks are developed for evaluating VOS performance, catering to

both single-object [Perazzi et al. 2016; F. Li et al. 2013; Ochs and Brox 2011; Lamdouar et al. 2020] and multi-object [Pont-Tuset et al. 2017; Xu et al. 2018] scenarios. Two major protocols are widely explored in VOS research, namely unsupervised [Z. Yang et al. 2019; C. Yang et al. 2021; X. Lu et al. 2019; Ventura et al. 2019; D. Cho et al. 2019; S. Li et al. 2018; Luiten et al. 2020; H. Lin et al. 2021] and semi-supervised VOS [Vondrick et al. 2018; Lai and W. Xie 2019; Lai et al. 2020; Miao et al. 2022; X. Wang et al. 2019; Jabri et al. 2020; Caron et al. 2021; S. W. Oh et al. 2019; Z. Yang and Y. Yang 2022; H. K. Cheng and Schwing 2022; Pan et al. 2022]. Notably, the term "unsupervised" exclusively indicates that no groundtruth annotation is used *during inference time* (*i.e.,* no inference-time supervision). In contrast, the semi-supervised VOS employs first-frame groundtruth annotations to initiate the object tracking and mask propagation in subsequent frames. This paper focuses on unsupervised VOS and utilises motion as a crucial cue for object discovery.

Motion segmentation focuses on discovering objects through their movement and generating corresponding segmentation masks. Existing benchmarks for motion segmentation largely overlap with those used for VOS evaluation, especially in the single-object case. For multi-object motion segmentation, datasets [J. Xie et al. 2022; J. Xie et al. 2024] have been specifically curated from VOS benchmarks to exclusively focus on sequences with dominant locomotion. There are two major setups in the motion segmentation literature: one that relies on motion information *only* to distinguish moving elements from the background through spatial clustering [C. Yang et al. 2021; Meunier et al. 2022; Meunier and Bouthemy 2023] or explicit supervision [Lamdouar et al. 2021; J. Xie et al. 2022]; the other [Bideau and Learned-Miller 2016b; Mahendran et al. 2018b; Y. Yang et al. 2021; Choudhury et al. 2022; J. Xie et al. 2024; Ponimatkin et al. 2023] that enhances motion-based object discovery by incorporating appearance information. We term these two approaches "flow-only" and "RGB-based" segmentation, respectively, and explore both setups in this work.

Segment Anything Model (SAM) [Kirillov et al. 2023] has demonstrated impressive ability on image segmentation across various scenarios. It was trained on the SA-1B datasets with over one billion self-labelled masks and 11 million images. Such large-scale training renders it a strong zero-shot generalisability to unseen

domains. Many works adapt the SAM model to perform different tasks, such as tracking [Y. Cheng et al. 2023], change detection [Zheng et al. 2024], and 3D segmentation [Cen et al. 2023]. Some other works extend SAM towards more efficient models [C. Zhang et al. 2023; Zhao et al. 2023; Xiong et al. 2023], and more domains [J. Ma et al. 2024; Wu et al. 2023; Kirillov et al. 2023; T. Chen et al. 2023]. However, most studies follow the default prompt options in SAM (*i.e.,* points, boxes, and masks). Recent works [Zou et al. 2023; Y. Sun et al. 2024] have shown that more versatile prompts, including scribbles and visual references, can lead to improvements. In this paper, we explore a novel route that prompts SAM with optical flow and demonstrate its effectiveness for moving object segmentation.

4.3 SAM preliminaries

The Segment Anything Model (SAM) is engineered for high-precision image segmentation, accommodating both user-specified prompts and a fully autonomous operation mode. When guided by user input, SAM accepts various forms of prompts including points, boxes, masks, or textual descriptions to accurately delineate the segmentation targets. Alternatively, in its automatic mode, SAM uses points on a uniform grid as prompts, to propose all plausible segmentation masks that capture objects and their hierarchical subdivisions—objects, parts, and subparts. In this case, the inference is repeated for each prompt of the grid, generating masks for each prompt in turn, and the final mask selection is guided by the predicted mask IoU scores.

Architecturally, SAM is underpinned by three foundational components: (i) Image encoder extracts strong image features via a heavy Vision Transformer (ViT) backbone, which is pre-trained by the Masked Auto-Encoder (MAE) approach; (ii) The prompt encoder converts the input prompts into positional information which helps with locating the segmentation target; (iii) Mask decoder features a lightweight two-way transformer that takes in a combination of encoded prompt tokens, learnable mask tokens, and an IoU prediction token as input queries. These queries iteratively interact with the dense spatial features from image encoder, leading to the final mask predictions and IoU estimations. In the next sections, we describe two distinct, yet simple variants to effectively tailor SAM for motion segmentation.

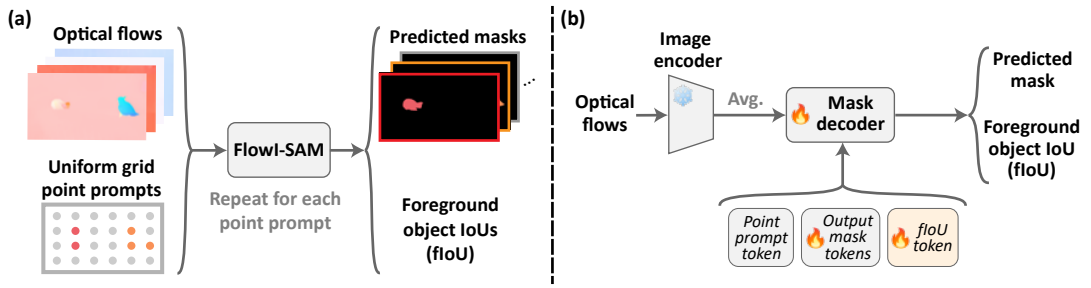


Figure 4.2: **Overview of `FlowI-SAM`. (a)** Inference pipeline of `FlowI-SAM`. **(b)** Architecture of `FlowI-SAM` with trainable parameters labelled. The point prompt token is generated by a frozen prompt encoder.

4.4 Frame-level segmentation I: flow as input

In this section, we focus on discovering moving objects from individual frames by exploiting motion information *only*, to yield corresponding segmentation masks. Formally, given the optical flow input $F_t \in \mathbb{R}^{H\times W\times 3}$ at frame t, we aim to predict a segmentation mask $M_t^i \in \mathbb{R}^{H\times W}$ together with a foreground object IoU (fIoU) score $s^i_{\text{fIoU},t} \in \mathbb{R}^1$ for each object i,

$$\{M_t^i, s^i_{\text{fIoU},t}\}_{i=0}^N = \Phi_{\texttt{FlowI-SAM}}(F_t) \tag{4.1}$$

To adapt SAM for this new task, we formulate `FlowI-SAM` ($\Phi_{\texttt{FlowI-SAM}}$) by finetuning it on optical **Flow** Inputs, and re-purpose the original IoU prediction head to instead predict the fIoU, as illustrated in Fig. 4.2b. By definition, fIoU is a scalar that measures the "objectness": fIoU is 0 if the mask belongs to the background, and equal to the IoU between predicted and GT object masks for foreground moving object masks. A high fIoU indicates the predicted mask corresponds to the entire object, while a low fIoU might suggest the mask is erroneous or only captures a small part of the object.

Flow inputs with multiple frame gaps. To mitigate the effect of noisy optical flow, *i.e.* complicated flow fields due to stationary parts, articulated motion, and object interactions, *etc.*, we consider multiple flow inputs $\{F_{t,g}\}$ with different frame gaps (*e.g.*, $g \in \{(1,\text{-}1), (2,\text{-}2)\}$) for both training and evaluation stages. These multi-gap flow inputs are **independently** processed by the image encoder to obtain dense spatial features $\{d_{t,g}\}$ at a lower resolution $h \times w$, which are then combined by averaging the spatial feature maps across different flow gaps, *i.e.*, $d_t = \text{Average}_g(\{d_{t,g}\}) \in \mathbb{R}^{h\times w\times d}$. These averaged spatial features are then

treated as keys and values in the mask decoder.

`FlowI-SAM` **inference.** To discover all moving objects from flow input, the model is prompted by points on a uniform grid. Each point prompt outputs a pair of mask and objectness score predictions. This mechanism is the same as in the original SAM formulation, and is illustrated in Fig. 4.2a. The final segmentation is selected using Non-Maximum Suppression (NMS) based on the predicted fIoU and overlap ratio.

`FlowI-SAM` **training.** To adapt the pre-trained SAM model for optical flow inputs, we finetune the lightweight mask decoder, while the image encoder and the prompt encoder remain frozen. The overall loss is formulated as:

$$\mathcal{L}_{\texttt{FlowI-SAM}} = \frac{1}{NT}\sum_{i,t}^{N,T}\left(\mathcal{L}_{\text{BCE}}(M_t^i, \hat{M}_t^i) + \lambda_f \|s_{\text{fIoU},t}^i - \hat{s}_{\text{fIoU},t}^i\|^2\right) \tag{4.2}$$

where $\hat{M}_t^i$ and $\hat{s}_{\text{fIoU},t}^i$ denote the groundtruth segmentation masks and fIoU, and λ_f is a scale factor.

4.5 Frame-level segmentation II: flow as prompt

In this section, we adapt SAM for video object segmentation by processing RGB frames, with optical flow as a prompt. We term this frame-level segmentation architecture `FlowP-SAM` for **Flow** as **P**rompt SAM. As shown in Fig. 4.3b, `FlowP-SAM` encompasses two major modules, namely the flow prompt generator and the segmentation module. The flow prompt generator takes optical flow as inputs, and produces flow prompts that can be used as supplemental queries to infer frame-level segmentation masks M_t^i from RGB inputs I_t. Formally,

$$\{M_t^i,\, s_{\text{fIoU},t}^i,\, s_{\text{MOS},t}^i\}_{i=0}^N = \Phi_{\texttt{FlowP-SAM}}(F_t, I_t) \tag{4.3}$$

where $s_{\text{MOS},t}^i$ indicates the moving object score (MOS) predicted by the flow prompt generator, while $s_{\text{fIoU},t}^i$ denotes the foreground object IoU (fIoU) estimated by the segmentation module. Specifically, MOS measures whether each input point prompt (therefore the resultant mask) is within a moving object region based on observing flow fields. Groundtruth MOS scores are binary (*i.e.*, $\hat{s}_{\text{MOS},t}^i = 1$ if the point prompt is within GT annotation, and $\hat{s}_{\text{MOS},t}^i = 0$ otherwise). On the other

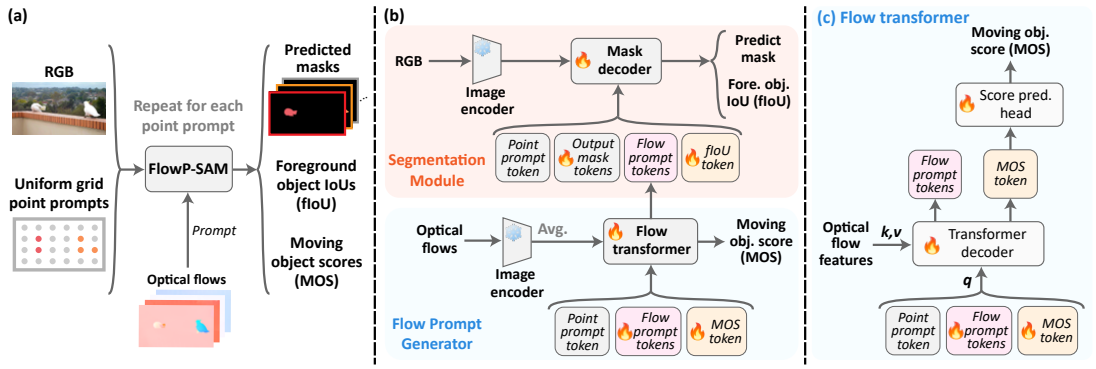


Figure 4.3: **Overview of `FlowP-SAM`. (a)** Inference pipeline of `FlowP-SAM`. **(b)** Architecture of `FlowP-SAM`. The flow prompt generator produces flow prompts to be injected into a SAM-like RGB-based segmentation module. Both modules take in the same point prompt token, which is obtained from a frozen prompt encoder. **(c)** Detailed architecture of the flow transformer. The input tokens function as queries within a lightweight transformer decoder, iteratively attending to dense flow features. The output moving object score (MOS) token is then processed by an MLP-based head to predict a score indicating whether the input point prompt corresponds to a moving object.

hand, fIoU follows the same formulation as in `FlowI-SAM`, *i.e.,* predicting IoUs for foreground objects and yielding 0 for background regions.

Flow prompt generator consists of (i) a frozen SAM image encoder, where the dense spatial features are extracted from optical flow inputs at different frame gaps, followed by an averaging across frame gaps; (ii) a flow transformer, with the detailed architecture depicted in Fig. 4.3c, where we first stack the input point prompt (*i.e.,* a positional embedding) with learnable flow prompts and moving object score (MOS) tokens to form queries. These queries then iteratively attend to dense flow features in a lightweight transformer decoder. There are two outputs from the flow prompt generator, namely, the refined flow prompts, and an MOS token, which is subsequently processed by an MLP-based head to yield a final moving object score.

Segmentation module has a structure that resembles the original SAM, except for two adaptions: (i) The IoU-prediction head is re-purposed to predict foreground object scores (fIoU) (same as the `FlowI-SAM`); (ii) The outputs tokens from flow prompt generator are injected as additional query inputs.

`FlowP-SAM` inference. Similar to `FlowI-SAM`, we prompt `FlowP-SAM` with single point prompts from a uniform grid to iteratively predict possible segmentation masks, together with MOS and fIoU estimations. These predicted scores are av-

eraged, *i.e.,* $(s^i_{\text{MOS},t} + s^i_{\text{fIoU},t})/2$, and then utilised to guide post-processing which involves the NMS and overlay of selected masks.

`FlowP-SAM` **training.** We train `FlowP-SAM` in an end-to-end fashion while keeping the SAM pre-trained prompt encoder and image encoders frozen. The flow transformer is trained from scratch,

$$\begin{aligned}\mathcal{L}_{\texttt{FlowP-SAM}} =& \frac{1}{NT}\sum_{i,t}^{N,T}\Big(\mathcal{L}_{\text{BCE}}(M^i_t, \hat{M}^i_t) \\ &+\lambda_m \mathcal{L}_{\text{BCE}}(s^i_{\text{MOS},t}, \hat{s}^i_{\text{MOS},t}) + \lambda_f \|s^i_{\text{fIoU},t} - \hat{s}^i_{\text{fIoU},t}\|^2\Big)\end{aligned} \tag{4.4}$$

where $\hat{M}^i_t$ corresponds to the groundtruth mask. $\hat{s}^i_{\text{MOS},t}$ and $\hat{s}^i_{\text{fIoU},t}$ indicate the groundtruth of two predicted scores, with λ_m and λ_f being the scale factors.

4.6 Sequence-level mask association

In this section, we outline our method for linking the frame-wise predictions for each moving object into a continuous track throughout the sequence. Specifically, we compute two types of masks: *frame-wise* masks M at the current frame using the model (`FlowI-SAM` and/or `FlowP-SAM`); and *sequence-level* masks $\mathcal{M}$, that are obtained by propagating the initial frame prediction with optical flow, we then update the mask of the current frame by making a comparison between the two. The following section details our update mechanism.

Update mechanism. This process aims to associate object masks across frames, as well as to determine whether the sequence-level results at a particular frame should be obtained directly from frame-level predictions at that frame or by propagating from previous frame results.

Specifically, given a sequence-level mask for object i at frame $t-1$ (*i.e.,* $\mathcal{M}^i_{t-1}$), we first warp it to frame t using optical flow,

$$\mathcal{M}^i_{t\leftarrow t-1} = \text{warp}(\mathcal{M}^i_{t-1}, F_{t-1}) \tag{4.5}$$

We then consider three sets of masks: (i) the warped masks $\{\mathcal{M}^i_{t\leftarrow t-1}\}$; (ii) the

frame-level predictions $\{M_t^i\}$ at frame t; and (iii) the frame-level predictions from neighboring frames (with Δt gap) after aligning them to the current frame using optical flow (*i.e.,* $\{M_{t\leftarrow t+\Delta t}^i\}$). For each pair of mask sets, we perform a pairwise Hungarian matching based on the IoU score, resulting in three pairings in total. The Hungarian matched pairs can then reflect the *temporal consistency* across these predictions based on the *transitivity* principle, i.e. f object i in (i) matches with object j in (ii) and object k in (iii), then the latter two objects must also match with each other. If such transitivity holds, we set the consistency score $c_i = 1$, and $c_i = 0$ otherwise.

This matching process is repeated for $\Delta t \in \{1, 2, -1, -2\}$, resulting in an averaged consistency score $\bar{c}_i$, which guides the following mask update:

$$\mathcal{M}_t^i = \begin{cases} M_t^i & \text{if } \bar{c}_i \geq 0.5 \\ \mathcal{M}_{t\leftarrow t-1}^i & \text{if } \bar{c}_i < 0.5 \end{cases} \tag{4.6}$$

where $\mathcal{M}_t^i$ denotes the sequence-level mask prediction for object i at frame t.

The rationale behind this is that the two methods have their own strengths and drawbacks: propagation is safe in preserving the object identity, but the mask quality degrades over time, whereas updating ensures high-quality masks but comes with the risk of mis-associating object identities. Thus, if the current frame-wise mask is temporally consistent, then we can be reasonably confident to update the mask, if not, then we choose the safer option and propagate the previous mask.

Note, we do this separately for each object $i \in N$, which gets updated or propagated independently. We lastly layer all objects back together (to its original order) and remove any overlaps to obtain the final sequence-level predictions.

4.7 Experiments

4.7.1 Datasets

Single-object benchmarks. For single-object motion segmentation, we adopt standard datasets, including DAVIS2016 [Perazzi et al. 2016], SegTrackv2 [F. Li et al. 2013], FBMS-59 [Ochs et al. 2014], and MoCA [Lamdouar et al. 2020]. Al-

though SegTrackv2 and FBMS-59 include a few multi-object sequences, following the common practice [C. Yang et al. 2021; Lamdouar et al. 2021], we treat them as single-object benchmarks by grouping all moving objects into a single foreground mask. MoCA stands for Moving Camouflaged Animals, designed as a camouflaged object detection benchmark. Following [C. Yang et al. 2021; Lamdouar et al. 2021; J. Xie et al. 2022], we adopt a filtered MoCA dataset by excluding videos with predominantly no locomotion.

Multi-object benchmarks. In terms of multi-object segmentation, we report the performance on DAVIS2017 [Pont-Tuset et al. 2017], DAVIS2017-motion [Pont-Tuset et al. 2017; J. Xie et al. 2022], and YouTube-VOS2018-motion [Xu et al. 2018; J. Xie et al. 2024], where DAVIS2017 is characterised by predominantly moving objects, each annotated as distinct entities. For example, a man riding a horse would be separately labelled. In contrast, DAVIS2017-motion re-annotates the objects based on their joint movements such that objects with shared motion are annotated as a single entity. For example, a man riding a horse is annotated as a single entity due to their shared motion.

The YouTubeVOS2018-motion [J. Xie et al. 2024] dataset is a curated subset of the original YouTubeVOS2018 [Xu et al. 2018]. It specifically excludes video sequences involving common motion, severe partial motion, and stationary objects, making it ideally suited for motion segmentation evaluation. Whereas, the original dataset also annotates many stationary objects and only provides partial annotations for a subset of moving objects.

Summary of evaluation datasets. To investigate the role of motion in object discovery and segmentation, we adopt all aforementioned benchmarks, which consist of only predominantly moving object sequences. Notably, for the evaluation of `FlowI-SAM`, we exclude the class-labelled DAVIS17 dataset, as commonly moving objects cannot be separated solely based on motion cues.

Training datasets. To adapt the RGB pre-trained SAM for moving object discovery and motion segmentation, we train both `FlowI-SAM` and `FlowP-SAM` first on the synthetic dataset introduced by [J. Xie et al. 2022], as described below, and then on real-world video datasets, including DAVIS16, DAVIS17, and DAVIS17-m.

4.7.2 Evaluation metrics

To assess the accuracy of predicted masks, we report intersection-over-union ($\mathcal{J}$), except for MoCA where only the ground-truth bounding boxes are given, we instead follow the literature [C. Yang et al. 2021] and report the detection success rate (SR). Regarding multi-object benchmarks, we additionally report the contour accuracy ($\mathcal{F}$) in the Appendix.

In this work, we differentiate between the frame-level and sequence-level methods, and adopt two distinct evaluation protocols: (i) Since frame-level methods generate the segmentation *independently* for each frame, we apply per-frame Hungarian matching to match the object masks between predictions with the groundtruth, before the evaluation; (ii) Conversely, sequence-level methods employ an extra step to link object masks across frames. As a result, the Hungarian matching is conducted globally for each sequence, *i.e.,* the object IDs between predicted and groundtruth masks are matched once per sequence. Given the added complexity and potential errors during frame-wise object association, sequence-level predictions are often considered a greater challenge.

4.7.3 Implementation details

In this section, we summarise the experimental setting in our frame-level segmentation models. For more information regarding detailed architectures, hyperparameter settings, and sequence-level mask associations, please refer to the Appendix.

Flow computation. We adopt an off-the-shelf method (RAFT [Teed and Deng 2020]) to estimate optical flow with multiple frame gaps at (1,-1) and (2,-2), except for YTVOS18-m and FBMS-59, where higher frame gaps at (3,-3) and (6,-6) are used to compensate for slow motion. Following common practice [C. Yang et al. 2021; J. Xie et al. 2022], we convert optical flow into 3-channel RGB format using a standard color wheel [Baker et al. 2007].

Model settings. For both `FlowI-SAM` and `FlowP-SAM`, we follow the default SAM setting and adopt the first output mask token (out of four) for mask predictions. For `FlowI-SAM`, we deploy two versions of the pre-trained SAM image encoder, specifically ViT-B and ViT-H, to extract optical flow features.

Regarding `FlowP-SAM`, for efficiency reasons, we utilise ViT-B to encode optical

flows and employ ViT-H as the image encoder for RGB frames. We initialise the flow prompt generator with 4 learnable flow prompt tokens, which are subsequently processed by a light-weight two-layer transformer decoder in the flow transformer module.

Evaluation settings. At inference time, for `FlowI-SAM` with flow-only inputs, we input independent point prompts over a 10×10 uniform grid, while for `FlowP-SAM`, to take account for more complicated RGB textures, we consider a large grid size of 20×20.

Mask selection over multiple point prompts. During post-processing, we utilise the predicted scores (fIoU for `FlowI-SAM`, and an average of MOS and fIoU for `FlowP-SAM`) as guidance throughout the mask selection process: (i) Non-Maximum Suppression (NMS) is applied to filter out repeating masks and keep the ones with higher scores; (ii) The remaining masks are then ranked according to their scores and top-n masks are retained ($n = 5$ for `FlowI-SAM`, and $n = 10$ for `FlowP-SAM`); (iii) These n masks are overlaid by allocating masks with higher scores at the front.

Training settings. The training is performed in two stages, which involves synthetic pre-training on the dataset proposed by [J. Xie et al. 2022], followed by finetuning on the real DAVIS sequences, as detailed in the Appendix. YTVOS is not used for fine-tuning as there is only a low proportion of moving object sequences. We train both models in an end-to-end manner using the Adam Optimiser at a learning rate of $3e^{-5}$. The training was conducted on a single NVIDIA A40 GPU, with each mode taking roughly 3 days to reach full convergence.

4.7.4 Ablation study

In this section, we present a series of ablation studies on key designs in the per-frame `FlowP-SAM` model. We refer the reader to the Appendix for a more detailed ablation analysis on the designs in `FlowI-SAM` and `FlowP-SAM` models, as well as in our sequence-level method.

Ablation studies for `FlowP-SAM`. As illustrated in Tab. 4.1, we start from the vanilla SAM and progressively add our proposed components. Note that, we adopt the same inference pipeline (*i.e.,* the same point prompts and post-processing

Stage		Predicted scores for post-processing	Flow prompt	FT mask decoder	DAVIS17 $\mathcal{J}\uparrow$	DAVIS16 $\mathcal{J}\uparrow$
		IoU	✗	✗	25.2	30.3
+	Train flow prompt generator	IoU	✓	✗	61.9	80.6
		(MOS+IoU)/2	✓	✗	63.7	81.4
+	Finetune segment -ation module	(MOS+IoU)/2	✓	✓	65.5	81.5
		(MOS+fIoU)/2	✓	✓	69.9	86.1

Table 4.1: **Ablation analysis of `FlowP-SAM`.** The study starts from the vanilla SAM checkpoint and progressively introduces new components (labelled in blue). "MOS" is short for the moving object score, and "fIoU" indicates the foreground object IoU. The results are shown for frame-level predictions.

steps) for all predictions shown. Since the foreground object IoU (fIoU) is not predicted by the vanilla SAM, we instead apply default IoU predictions to guide the mask selection.

We *train the flow prompt generator* to simultaneously predict flow prompt tokens and moving object scores (MOS). The injection of flow prompts into the standard RGB-based SAM architecture results in notable enhancements, verifying the value of motion information for accurately determining object positions and shapes. Additionally, employing MOS as additional post-processing guidance yields further improvements.

Upon *finetuning the segmentation module*, we observe a slight enhancement in performance. Finally, substituting the default IoU predictions with fIoU scores achieves more precise mask selection, as evidenced by the improved results.

Discussion on the effectiveness of MOS and fIoU. As outlined in Sect. 4.3, the original SAM framework over-segments images into objects, parts, and sub-parts, which cannot be further distinguished using the default SAM IoU estimations. To adapt this setup for *object-only* discovery, we propose two new scores, *i.e.,* MOS and fIoU, as new criteria to filter out non-object masks. These scores effectively assess the "objectness" of masks: MOS determines if the predicted masks represent moving objects, while fIoU evaluates whether the masks depict complete objects and are not background segments. The effectiveness of this adaptation is validated in Table 4.1, where replacing IoU estimations with MOS + fIoU scores leads to noticeable performance boosts.

Model	Flow	RGB	Multi-object benchmarks			Single-object benchmarks			
			YTVOS18-m $\mathcal{J}\uparrow$	DAVIS17-m $\mathcal{J}\uparrow$	DAVIS17 $\mathcal{J}\uparrow$	DAVIS16 $\mathcal{J}\uparrow$	STv2 $\mathcal{J}\uparrow$	FBMS $\mathcal{J}\uparrow$	MoCA SR $\uparrow$
Flow-only methods									
COD [Lamdouar et al. 2020]	✓	✗	–	–	–	65.3	–	–	0.236
†MG [C. Yang et al. 2021]	✓	✗	37.0	38.4	–	68.3	58.6	53.1	0.484
†EM [Meunier et al. 2022]	✓	✗	–	–	–	69.3	55.5	57.8	–
`FlowI-SAM` **(ViT-B)**	✓	✗	56.7	63.2	–	**79.4**	69.0	72.9	**0.628**
`FlowI-SAM` **(ViT-H)**	✓	✗	**58.6**	**65.7**	–	79.1	**70.1**	**75.1**	0.625
RGB-based methods									
†VideoCutLER [X. Wang et al. 2023]	✗	✓	59.0	57.4	41.7	–	–	–	–
†Safadoust et al. [Safadoust and Güney 2023]	✗	✓	–	59.3	–	–	–	–	–
MATNet [T. Zhou et al. 2020]	✓	✓	–	–	56.7	82.4	50.4	76.1	0.544
DystaB [Y. Yang et al. 2021]	✗	✓	–	–	–	82.8	74.2	75.8	–
AMC-Net [S. Yang et al. 2021]	✓	✓	–	–	–	84.5	–	76.5	–
TransportNet [K. Zhang et al. 2021]	✓	✓	–	–	–	84.5	–	78.7	–
TMO [S. Cho et al. 2023]	✓	✓	–	–	–	85.6	–	79.9	–
`FlowP-SAM`	✓	✓	76.9	78.5	69.9	86.1	83.9	87.9	**0.645**
`FlowP-SAM+FlowI-SAM`	✓	✓	**77.4**	**80.0**	**71.6**	**86.2**	**84.2**	**88.7**	**0.645**

Table 4.2: **Frame-level comparison on video object segmentation benchmarks.** "†" indicates models that are trained without human annotations. For the results in the last row, we combine frame-level predictions from `FlowP-SAM` and `FlowI-SAM` (ViT-H).

4.7.5 Quantitative results

Given the distinct evaluation protocols outlined in Section 4.7.2, we report our method separately, with a frame-level analysis for `FlowI-SAM` (introduced in Section 4.4) and `FlowP-SAM` (introduced in Section 4.5), followed by a sequence-level evaluation.

Frame-Level Performance. Table 4.2 distinguishes between flow-only and RGB-based methods, where the former adopts optical flow as the only input modality, and the latter takes in RGB frames with optional flow inputs. Note that, the performance for some recent self-supervised methods is also reported, owing to the lack of the supervised baselines.

For flow-only segmentation, our `FlowI-SAM` (with both SAM image encoders) outperforms the previous methods by a large margin (>10%). For RGB-based segmentation, our `FlowP-SAM` also achieves state-of-the-art performance, particularly excelling at multi-object benchmarks. By combining these two frame-level predictions (`FlowI-SAM+FlowP-SAM`), we observe further performance boosts. This suggests the complementary roles of the flow and RGB modalities in frame-level segmentation, particularly when there are multiple moving objects involved. In particular, we show that using both models in tandem by layering `FlowI-SAM`'s predictions behind that of `FlowP-SAM` allows the model to fill in on missed predic-

Model	Flow	RGB	Multi-object benchmarks			Single-object benchmarks			
			YTVOS18-m $\mathcal{J}\uparrow$	DAVIS17-m $\mathcal{J}\uparrow$	DAVIS17 $\mathcal{J}\uparrow$	DAVIS16 $\mathcal{J}\uparrow$	STv2 $\mathcal{J}\uparrow$	FBMS $\mathcal{J}\uparrow$	MoCA SR $\uparrow$
Flow-only methods									
†SIMO [Lamdouar et al. 2021]	✓	✗	–	–	–	67.8	62.0	–	0.566
†Meunier et al. [Meunier and Bouthemy 2023]	✓	✗	–	–	–	73.2	55.0	59.4	–
†OCLR [J. Xie et al. 2022]	✓	✗	46.5	54.5	–	72.1	67.6	70.0	0.599
OCLR-real [J. Xie et al. 2022]	✓	✗	49.5	55.7	–	73.3	65.9	70.5	0.605
`FlowI-SAM` **(seq, ViT-B)**	✓	✗	51.9	60.0	–	**78.4**	66.9	69.0	**0.615**
`FlowI-SAM` **(seq, ViT-H)**	✓	✗	**53.8**	**61.5**	–	78.0	**67.7**	**71.5**	0.604
RGB-based methods									
UnOVOST [Luiten et al. 2020]	✗	✓	–	–	66.4	–	–	–	–
Propose-Reduce [H. Lin et al. 2021]	✗	✓	–	–	67.0	–	–	–	–
OCLR-flow [J. Xie et al. 2022] + SAM	✓	✓	57.0	62.0	–	80.6	71.5	79.2	–
PMN [M. Lee et al. 2023]	✓	✓	–	–	–	85.6	–	77.8	–
Xie et al. [J. Xie et al. 2024] + SAM	✓	✓	71.1	70.9	–	86.6	**81.3**	**85.7**	–
DEVA [H. K. Cheng et al. 2023]	✗	✓	–	–	70.4	87.6	–	–	–
UVOSAM [Z. Zhang et al. 2024]	✗	✓	–	–	**77.5**		–	–	–
`FlowP-SAM+FlowI-SAM` **(seq)**	✓	✓	**74.7**	**74.3**	71.0	**87.7**	80.1	82.8	**0.647**

Table 4.3: **Sequence-level comparison on video object segmentation benchmarks.** "†" indicates models that are trained without human annotations. "seq" indicates that our sequence-level predictions with object masks matched across frames. We adopt `FlowP-SAM` and `FlowI-SAM` (ViT-H) to obtain the results in the last row.

tions (such as motion blur, poor lighting, or small objects).

Sequence-Level Performance. For flow-based segmentation, we apply the mask association technique introduced in 4.6 to obtain sequence-level predictions from per-frame `FlowI-SAM` results. To ensure a fair comparison, we additionally finetune the synthetic-trained OCLR [J. Xie et al. 2022] model on the real-world dataset (DAVIS) with groundtruth annotations provided, resulting in "OCLR-real" results. As shown by the top part of Table 4.3, `FlowI-SAM` (seq) demonstrates superior performance against OCLR-real, benefiting from the robust prior knowledge in pre-trained SAM.

For RGB-based segmentation, we obtain our sequence-level predictions by `FlowI-SAM+FlowP-SAM`. As shown in the lower part of Table 4.3, our method achieves outstanding performance across single- and multi-object benchmarks. Note, DAVIS17 annotations are class-based, which could be unclear and inconsistent with the class-agnostic unsupervised VOS methods. More discussion and results are provided in the Appendix.

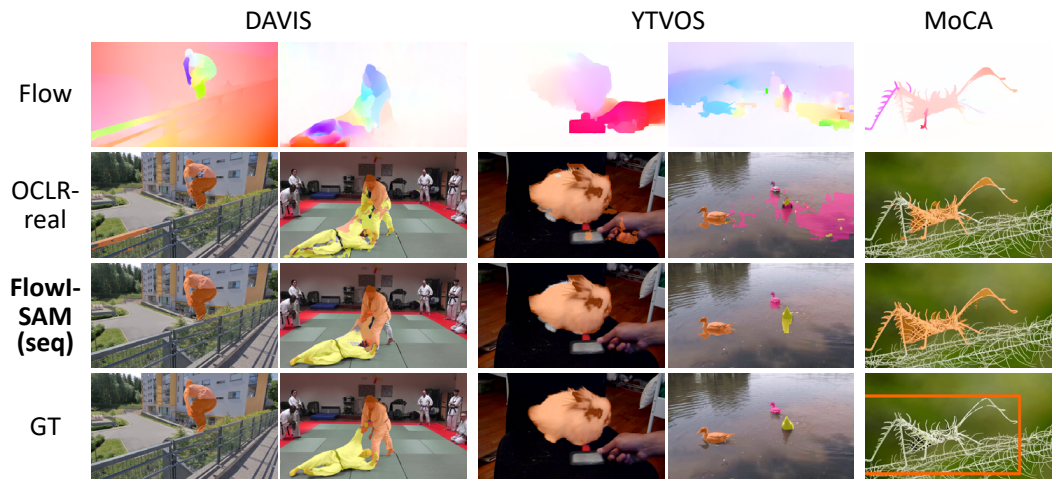


Figure 4.4: **Qualitative comparison of flow-only segmentation methods** on DAVIS (left), YTVOS (middle), and MoCA (right) sequences. Our `FlowI-SAM` (seq) successfully identifies moving objects from noisy optical flow background (*e.g.,* the ducks in the fourth column).

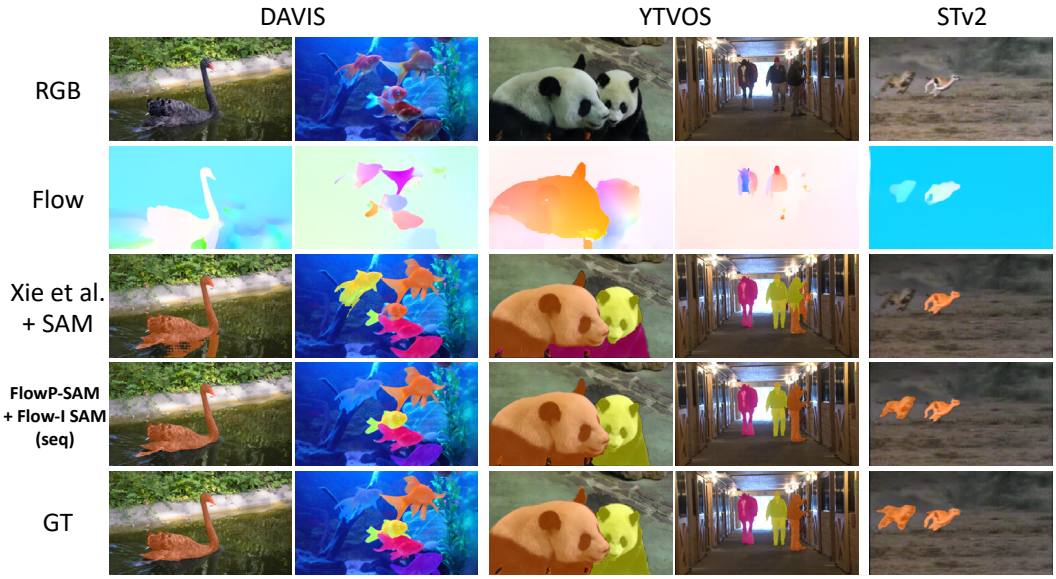


Figure 4.5: **Qualitative comparison of RGB-based segmentation methods** on DAVIS (left), YTVOS (middle), and SegTrackv2 (right). While the previous method (the third row) struggles to disentangle multiple moving objects (*e.g.,* mixed gold fishes in the second column), our `FlowP-SAM`+`FlowI-SAM` (seq) accurately separates and segments all moving objects.

4.7.6 Qualitative visualisations

In this section, example visualisations are provided across multiple datasets. We refer more visualisations to the Appendix.

Fig. 4.4 illustrates the segmentation predictions based on only optical flow inputs. Compared to OCLR-real, our `FlowI-SAM` accurately identifies and disentangles the moving objects from the noisy backgrounds (*e.g.,* the person in the first column

and the ducks in the fourth column), as well as extracts fine structures (*e.g.,* the camouflaged insect in the fifth column) from optical flow.

Fig. 4.5 further provides the visualisations of the RGB-based method, where the prior work (Xie et al. [J. Xie et al. 2024] + SAM [Kirillov et al. 2023]) sometimes fails to (i) identify the moving objects (*e.g.,* missing leopard in the fifth column); (ii) distinguish between multiple objects (*e.g.,* entangled object segmentation in the second and fourth columns), while our `FlowI-SAM`+`FlowP-SAM` (seq) incorporates RGB-based prediction with flow prompts, resulting in the accurate localisation and segmentation of moving objects.

4.8 Discussion

In this paper, we focus on moving object segmentation in real-world videos, by incorporating per-frame SAM with motion information (optical flow) in two ways: (i) for flow-only segmentation, we introduce `FlowI-SAM` that directly takes in optical flow as inputs; (ii) for RGB-based segmentation (`FlowP-SAM`), we utilise motion information to generate flow prompts as guidance. The former (`FlowI-SAM`) is particularly effective in scenarios with predominant motion and/or where RGB information might introduce confusion, such as in moving object detection and camouflaged object discovery. Additionally, owing to simple texture and a low cross-domain gap in optical flow, it generalises to diverse domains beyond everyday videos. On the other hand, `FlowP-SAM` focuses on common videos where both RGB and motion are informative and can be utilised to resolve ambiguities for independent objects in common motion.

Both approaches deliver state-of-the-art performance in frame-level segmentation across single- and multi-object benchmarks. Additionally, we develop a frame-wise association method that amalgamates predictions from `FlowI-SAM` and `FlowP-SAM`, achieving sequence-level segmentation predictions that outperform existing methods on DAVIS16, DAVIS17-m, YTVOS18-m, and MoCA.

The major limitation of this work is its extended running time, attributed to the computationally heavy image encoder in the vanilla SAM. However, our approach is generally applicable to other prompt-based segmentation models. With the emergence of more efficient versions of SAM, we anticipate a significant reduction

in inference time.

Acknowledgments

This research is supported by the UK EPSRC CDT in AIMS (EP/S024050/1), a Clarendon Scholarship, a Royal Society Research Professorship RP\R1\191132, and the UK EPSRC Programme Grant Visual AI (EP/T028572/1).

Appendices

The appendices can be found on the online version of the paper.

Statement of authorship

A statement of authorship for this paper is provided in the Appendix.

Part II

$x(t)$ – Learning from change patterns over time

Chapter 5

It's About Time: Analog clock reading in the wild

The paper was published at the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022.

timeline / time series / time after time

Generated with Meta AI

It's About Time:
Analog clock reading in the wild

Charig Yang[1] Weidi Xie[1,2] Andrew Zisserman[1]

[1]VGG, Department of Engineering Science, University of Oxford

[2]Shanghai Jiao Tong University

{charig,weidi,az}@robots.ox.ac.uk

https://www.robots.ox.ac.uk/~vgg/research/time

Abstract

In this paper, we present a framework for reading analog clocks in natural images or videos. Specifically, we make the following contributions: *First*, we create a scalable pipeline for generating synthetic clocks, significantly reducing the requirements for the labour-intensive annotations; *Second*, we introduce a clock recognition architecture based on spatial transformer networks (STN), which is trained end-to-end for clock alignment and recognition. We show that the model trained on the proposed synthetic dataset generalises towards real clocks with good accuracy, advocating a Sim2Real training regime; *Third*, to further reduce the gap between simulation and real data, we leverage the special property of "time", *i.e.* uniformity, to generate reliable pseudo-labels on real unlabelled clock videos, and show that training on these videos offers further improvements while still requiring zero manual annotations. *Lastly*, we introduce **three** benchmark datasets based on COCO, Open Images, and The Clock movie, with full annotations for time, accurate to the minute.

5.1 Introduction

Humans are able to sense the time to some level of granularity given environmental cues, such as luminance or the extent of shadows. However, in order to know the *exact* time we read from a time-keeping instrument such as a clock or watch. Clocks come in different shapes, forms and styles, and humans are able to read them despite not having seen the particular clock before. In this paper our objective is to enable a machine to perform the same task of telling the time from clocks *in the wild*.

Nowadays, clocks come in two main types – digital and analog. Digital clocks can be handled by text spotting methods [Lyu et al. 2018; Liao et al. 2021; Y. Liu et al. 2020; X. Liu et al. 2018; Qin et al. 2019] with relative ease, which we show in the Appendix, but reading analog clocks is a different and challenging problem: there are significant appearance variations between clock faces (see Figures 5.2 and 5.5), their imaged shape and the position of the numbering is severely affected by camera viewpoint, and the presence of shadows and specular reflections add confusion with the clock hands. While this problem has existed for a long time [Duvallet 2016; Vara 2021; Al-Baz 2020], no previous solutions are able to robustly read the time from clocks, apart from under extremely limited situations. And, somewhat surprisingly, reading analog clocks in unconstrained images has been largely overlooked in the computer vision literature. Additionally, there are no reliable benchmarks for evaluation, hindering the research community from tackling this task.

However, there are similarities between analog clock reading and text spotting in natural scenes – since in both cases the design (of the clock face or text font) is chosen to be readable, and in both cases there is a detection stage and then a reading stage. Clock reading has the additional challenges outlined above, but it also has an additional redundancy cue in that the position of the hour hand gives some information about the position of the minute hand. Given the similarities between the two tasks, we start with an approach that has been successful for text spotting: using synthetic datasets [Gupta et al. 2016; Jaderberg et al. 2014] and spatial transformer networks [B. Shi et al. 2016], and ask if these ideas transfer to our task. We find that they do to an extent, and provide further contributions to

bridge the Sim2Real generalisation gap.

While being able to carry out a new task is a sufficient reward in itself, there are a number of applications that are opened up, once we are able to automate time reading in images in the wild: first, it will now be possible to offer corrections where the image's EXIF metadata differs from the time read in the image; second, in video forensics, it will now be possible to spot if the video has been tampered with if the temporal ordering does not progress monotonically or if there is manipulation of the speed [Hosler and Stamm 2020]; third, it provides a new method of searching, retrieving and grouping images and videos; and, finally, clocks are just a (rather difficult) instance of an analog scale, and the methods we have proposed can be applied with a simple adaptation to other type of scales – from scientific instruments to industrial gauges.

In this paper, we provide the first working solution to these issues. We make the following contributions:

First, we propose a synthetic dataset generator, SynClock, that is designed to generalize to real clocks. SynClock has several controllable features that enables it to generate clocks with a wide range of designs. Moreover, we mimic difficulties faced in recognising real clocks into the generator's data augmentation process, e.g. homography transformation, artefacts, shadows.

Second, we design a two-stage framework involving detection and recognition stages. The detection can simply be an off-the-shelf object detection model. The recognition stage involves an alignment network, which is a spatial transformer network that regresses homography transformation parameters in order to make the clock fronto-parallel, and a classification network, which determines the time accurate to the minute. We show that the model is able to generalise towards real clocks with good accuracy.

Third, we leverage the uniformity of time – that it flows at a constant rate, in order to generate pseudo-labels on unlabelled clock videos. Specifically, we can be reasonably confident that the time labels in a video are correct if the rate of change of predicted time is constant throughout the video. To achieve this, we use a bundle adjustment algorithm to filter eligible videos and train on those with the pseudo labels. We also propose a dataset of 3,443 unlabelled clock time-lapse videos, and

show that learning from pseudo-labelled real data improves the performance. We will release the raw videos, as well as reliable automatic annotations for 1.5M frames across 2,511 videos.

Fourth, we propose three new benchmark datasets. The first two are based on existing datasets for object detection, namely COCO and OpenImages. We also introduce the Clock Movies dataset, based on the film The Clock (2010), which is a 24-hour montage of different movies featuring clocksOur model achieves 80.4%, 77.3% and 79.0% top-1 accuracy on each dataset respectively, marking the first time that analog clocks can be read successfully in unconstrained images.

5.2 Related work

Analog clock reading. While there exists blog posts and repositories, this task is still largely absent in the research literature, which may be because of the lack of proper benchmarks. Traditional methods use handcrafted methods such as edge or line detection algorithms to read clocks [Horvath 2021; El-Naggar et al. 2018; Hossam 2017], but they only work on simple, clean, artefact-free, fronto-parallel clocks. Given the absence of labelled data, existing works usually consider to train on synthetic clocks [Duvallet 2016; Vara 2021; Al-Baz 2020], but only go as far as testing their models on synthetic test sets. These models hence are not designed with generalisation in mind, and usually fail to read clocks in real scenes. In this paper, in addition to adopting a more challenging data simulation pipeline, we also leverage the uniformity property of time, and mix the training with data of timelapse analog clock videos. As a result, the gap between simulation and real-world images has been largely minimised in a self-supervised manner, *i.e.* without using manual annotations.

Dial reading. The closet kin to our work is on automatic dial or gauge meter readings. These two tasks face similar challenges in overcoming the effects of blur, glare, and reflections. The solutions proposed are somewhat similar, using neural networks [Alexeev et al. 2020], projective transforms [Bao et al. 2019], and virtual dataset generators [Cai et al. 2020; Howells et al. 2021], which work very well for gauges with known shape and style. These have applications towards reading

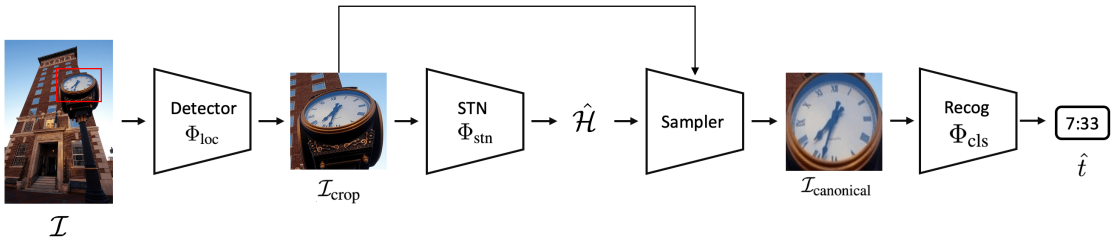


Figure 5.1: **Architecture.** Given an image $\mathcal{I}$, we first use an off-the-shelf object detector Φ_{loc} to obtain a cropped image $\mathcal{I}_{\text{crop}}$. We then pass the cropped image to a spatial transformer network Φ_{stn} which outputs a homography matrix $\mathcal{H}$, that can then be used to warp the cropped image to a canonical view $\mathcal{I}_{\text{canonical}}$. Lastly, the canonical image is passed to a classification network Φ_{cls} to predict the time.

electricity, water or gas dials [Salomon et al. 2020; Laroca et al. 2021; Naim et al. 2021; Alexeev et al. 2020]. Our task is more challenging as clock appearances vary greatly, and we further propose a method to bridge this generalisation gap.

Sim2Real transfer. In many computer vision and robotic tasks, synthetic datasets prove to be a useful source for training. This is especially true when the ground-truth is difficult or impossible to acquire at scale, including optical flow [Butler et al. 2012; Mayer et al. 2016], and text detection [Gupta et al. 2016] and recognition [Jaderberg et al. 2014], pose estimation [Doersch and Zisserman 2019], and motion segmentation [Lamdouar et al. 2021].

5.3 Architecture

In this paper, our goal is to train a computer vision system that can read the time shown on the analog clock from in-the-wild images. To achieve this, we propose a framework shown in Figure 5.1. Specifically, our proposed architecture takes an image as input, then sequentially undergoes *cropping*, *alignment*, and *reading*.

5.3.1 Clock localisation module (Φ_{loc})

Given an input image $\mathcal{I}$, we pass it through an object detection network, to localise and crop the clock.

$$\mathcal{I}_{\text{crop}} = \Phi_{\text{loc}}(\mathcal{I}; \Theta_{\text{loc}}) \in \mathbb{R}^{3 \times h_c \times w_c} \tag{5.1}$$

As we will show in the experiments in Section 5.7, the detection stage can be done

using an off-the-shelf object detector [Liang et al. 2021] without much impact on performance. This work hence focuses on the recognition stage.

5.3.2 Clock recognition module ($\Phi_{\mathbf{rec}}$)

The cropped clock could be passed directly to a classification network. However, this is usually not ideal for two reasons, firstly, due to the imperfections of the localisation module; and secondly, even if the clock is properly localised and cropped, it can still sometimes be difficult to read due to the viewpoint (as viewpoint alters angles). To overcome these problems, we adopt a spatial transformer network (STN) [Jaderberg et al. 2015] for *alignment*, facilitating the *recognittion*, *i.e.* $\Phi_{rec}(\cdot) = \Phi_{\text{cls}}(\Phi_{\text{stn}}(\cdot))$.

Spatial transformer network ($\Phi_{\mathbf{stn}}$) takes the cropped image ($\mathcal{I}_{\text{crop}}$) as input and outputs 8 homography transformation parameters:

$$\hat{\mathcal{H}} = \Phi_{\text{stn}}(\mathcal{I}_{\text{crop}}) \in \mathcal{R}^{3\times 3} \tag{5.2}$$

$$\mathcal{I}_{\text{canonical}} = \textsc{Sampler}(\mathcal{I}_{\text{crop}}, \hat{\mathcal{H}}) \in \mathcal{R}^{3\times h\times w} \tag{5.3}$$

where $\hat{\mathcal{H}} \in \mathcal{R}^{3\times 3}$ refers to the predicted homography transformation with 8 degree of freedoms, and $\textsc{Sampler}(\cdot)$ denotes a differentiable warping, that transform the cropped clock to its canonical view ($\mathcal{I}_{\text{canonical}}$), where the clock is fronto-parallel and the '12' position is at the top.

Classification network ($\Phi_{\mathbf{cls}}$). For reading the time, we quantise the time and pose the recognition problem as a 720-way classification, *i.e.* classifying both the hour (12) and minute (60) together. Specifically, we pass the canonical clock ($\mathcal{I}_{\text{canonical}}$) to a classification network, which outputs the probability for each class.

$$\hat{t} = \Phi_{\text{cls}}(I_{\text{canonical}}) \in \mathcal{R}^{720} \tag{5.4}$$

In the Appendix we compare this classification approach to time reading to a regression approach.

Each of the networks (Φ_{stn}, Φ_{cls}) is a standard ResNet-50 [He et al. 2016] pretrained

Figure 5.2: **Training data.** Left: example images from the SynClock dataset generator, which is designed to generate a wide range of clocks. We also add artefacts, such as random lines and shadows followed by augmentations. Right: example scenes from the Timelapse dataset, containing 3,443 unlabelled timelapse videos containing clocks. We train with this dataset using pseudo-labels with uniformity constraints.

on ImageNet [Deng et al. 2009].

After introducing the architecture, one question remains: *how can we efficiently train this clock recognition module, without a laborious annotation procedure?*

5.4 Synthetic data and Sim2Real training

To avoid laborious manual annotations, we describe a procedure for training alignment and recognition with simulated clocks, advocating a Sim2Real training regime. Specifically, the synthetic clocks are generated with different viewpoints, time, and styles.

5.4.1 Synthetic clock generator (SynClock)

Inspired by the idea for text spotting [Jaderberg et al. 2014], we propose a scalable pipeline for generating synthetic analog clocks. In order to generalise towards real clocks, we make the dataset sufficiently diverse with several controllable parameters, and add artefacts to simulate the real-world scenario. Specifically, we vary these parameters during training:

- **Background**: color.

- **Clock face**: size, shape, color.
- **Clock border**: thickness, color.
- **Tick marks**: whether to have ticks every minute or every hour, gap from border, length, thickness, color.
- **Numerals**: whether to have numerals, gap from border, font, font size, font thickness, color.
- **Hands**: whether to use 2, 3, or 4 hands (hour, minute, second, alarm), time, length, back length, thickness, color, whether to use arrow, arrow tip length, arrow size. (may be different for each hand)
- **Artefacts**: shadows beside a hand, random lines, random homography transformation.
- **Augmentation**: random blur, color channel jittering

Figure 5.2 shows examples of clocks produced by this SynClock generator, and further examples are given in the Appendix.

5.4.2 Training on SynClock

With full controls on the data generation procedure, we are able to to train both spatial transformer and the classification network. Specifically, Φ_{stn} and Φ_{cls} are trained with L1 loss and cross-entropy loss, using ground-truth transformation ($\mathcal{H}$) and time (t) generated from SynClock:

$$\mathcal{L} = \mathcal{L}_{\text{stn}} + \mathcal{L}_{\text{cls}} = \sum |\hat{\mathcal{H}} - \mathcal{H}| + \sum \hat{t} \log(t) \tag{5.5}$$

5.5 Pseudo-labelling real videos

While training on carefully designed synthetic clocks allows reasonable generalisation to real images, the domain gap between simulation and real images still exists, as will be demonstrated in our experiments in Section 5.7.3. In this section, we describe a simple idea for minimising the domain gaps by mixing the training with real video frames. One critical issue would be, *how can we obtain the ground-truth time for these video frames?*

5.5.1 Uniformity constraints

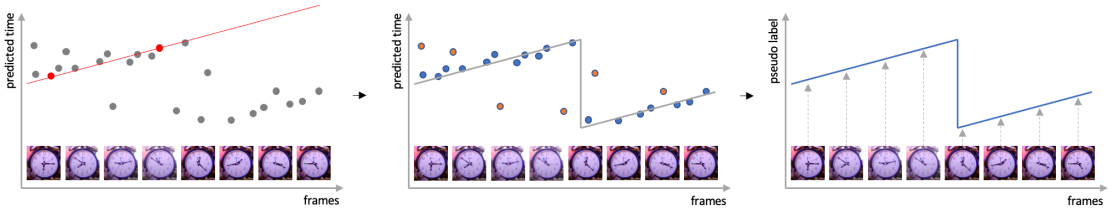


Figure 5.3: **Uniformity Constraints.** Given a timelapse video of a clock, we iteratively fit a line through randomly sampled predictions (left), before rectifying it into the valid range [0, 720) using the modulo operator and counting the inliers (middle). If the maximum inlier count is above a threshold, we then correct the readings using the fitted line (right) and add the pseudo-labelled clocks to the training set.

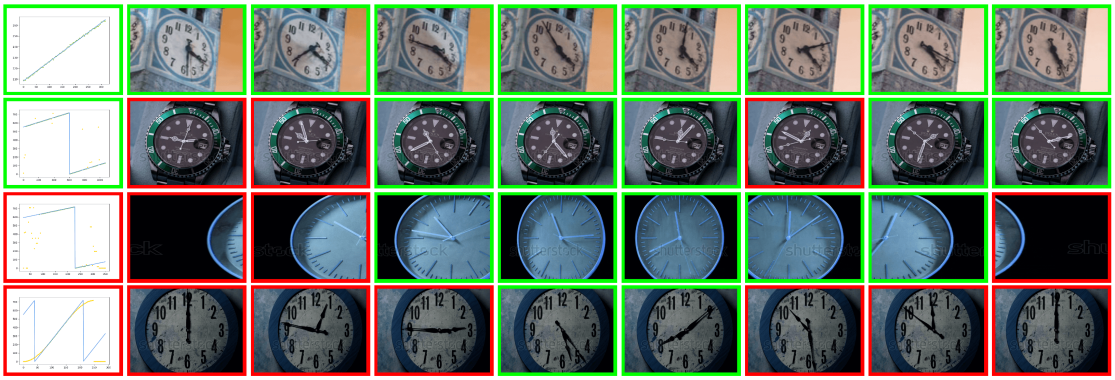


Figure 5.4: **Examples of filtering using uniformity constraints.** The first two rows show examples of videos that passed filtering, with incorrect predictions being calibrated accordingly. The bottom two rows show examples of videos that failed, due to out-of-frame and non-uniform speed respectively. The color of the box indicates success or failure.

We leverage the **uniformity** property of time, i.e. time flows unidirectionally at a constant rate in videos. Specifically, given an unlabelled clock video, we pass each individual frame through the localisation and recognition module to get the time predictions. As the recognition module has only been trained on synthetic data, it is likely to generate incorrect predictions for certain frames. Here, we can conveniently fit a line with RANSAC [Fischler and Bolles 1981] – a robust fitting algorithm that allows line fitting in the presence of outliers, incorrect time predictions in our case. An iteration of RANSAC involves fitting a line with the minimal number of randomly sampled points (two), counting points within a threshold distance as 'inliers', and the points that are distant from the line are treated as outliers. After running RANSAC multiple times, the best fitted line with maximum number of inliers are adopted, and all outliers can be re-calibrated accordingly, as shown in Figures 5.3 and 5.4.

Note that the data is cyclic, i.e. 11:59 (719) is connected to 0:00 (0). To overcome

this, we modified RANSAC such that it fits a sawtooth wave instead of a line, as seen in Figure 5.3. Specifically, after fitting a line through the randomly sampled points, if the predicted time is outside the [0, 720) range, we have to repeatedly add or subtract it by 720 so that it falls into the valid range, before counting the inliers. In practice, this can be implemented simply by applying a *modulo* (%) 720 operator on the fitted line.

5.5.2 Timelapse dataset

While the above method can work on any video with clocks running continuously, storing the videos of multiple hours long at scale would be impractical. Instead, we use *timelapse* clock videos, where time moves much faster, allowing video duration to be in the range of seconds and not hours. We hence collect a dataset of $3,443$ unlabelled time-lapse clock videos from the internet. Although no absolute time information can be extracted from these unlabelled videos, knowing that the speed is constant is sufficient for our use.

Joint training. After pseudo-labelling the videos, we select the ones with inlier proportion being above a threshold (fixed at 0.7), with a tight inlier margin of +-3 minutes. We also reject videos that move too slowly, that is, less than 10 minutes throughout the video. We add the videos that pass filtering to the training set to re-train the model. This whole process is automated, and hence no extra manual annotation or screening is required. Although the groundtruth transformation for aligning the clocks remains unavailable, meaning the intermediate regression loss only applies to the synthetic data, the model is still differentiable end-to-end.

$$\mathcal{L} = \mathcal{L}_{\text{stn}} + \mathcal{L}_{\text{cls}} = \sum_{\text{SynClock}} |\hat{\mathcal{H}} - \mathcal{H}| + \sum \hat{t} \log(t) \tag{5.6}$$

Iterative retraining. The process of pseudo-labelling and retraining can be performed iteratively. Specifically, after training with real data, the model becomes more capable of reading clocks, and hence can be used to generate better pseudo-labels to train the model. This process can be iterated for further performance improvements.

(a) COCO (b) OpenImages (c) Clock Movies

Figure 5.5: **Evaluation data.** From left to right, the figure shows example scenes and cropped clocks from COCO [T.-Y. Lin et al. 2014], OpenImages [Kuznetsova et al. 2020] and Clock Movies datasets. For the first two datasets, we filter scenes with readable analog clocks from the original datasets and provide time labels, while the latter is from 600 different movies. They contain 4,472 images in total and are used for evaluation.

5.6 Experimental setup

5.6.1 Datasets

We use separate datasets for training and testing. We utilise one synthetic and one pseudo-labelled real dataset for training, and hence are able to train the model with *zero* manual annotations. As this is a new task, there has not been proper benchmarking in the literature. We therefore propose three different test datasets. The summary statistics are given in Table 5.1. **All** five of these datasets are contributions of this paper.

Dataset	Type	Size	Split	Bbox	$\mathcal{H}$	Time
SynClock	image	∞	train	✗	✓	✓
Timelapse	video	3443	train	✗	✗	✗
COCO	image	1911	test	✓	✗	✓
OpenImages	image	1317	test	✓	✗	✓
Clock Movies	image	1244	test	✗	✗	✓

Table 5.1: **Statistics of the proposed datasets.** The first two datasets are used for training, while the other three for testing.

Iteration	Videos	Frames
Pseudo-label Round 1	1670	1.02M
Pseudo-label Round 2	2052	1.30M
Pseudo-label Round 3	2398	1.46M
Pseudo-label Round 4	2460	1.48M
Pseudo-label Round 5	2511	1.51M

Table 5.2: **Statistics of the Timelapse dataset.** This table shows the number of videos successfully labelled in each iteration of pseudo-labelling. Each iteration uses the model of the previous row.

Training

SynClock. We train our recognition module using the SynClock dataset, as previously explained. We also provide ground truth homographies to train the alignment network.

Timelapse. This dataset is a subset of WebVid dataset [Bain et al. 2021] used for video retrieval, using the "clocks time lapse" as the keyword for search query. It contains 3,443 unlabelled videos, and we train on this dataset using pseudo-labels, with the number of filtered videos shown in Table 5.2.

Testing

COCO. This dataset is a subset of COCO [T.-Y. Lin et al. 2014] dataset that contains clocks, with bounding boxes provided for 1,911 images and time manually labelled by the authors.

OpenImages. This dataset is a subset of OpenImages [Kuznetsova et al. 2020] dataset that contains clocks, with bounding boxes provided for 1,311 images and time manually labelled by the authors.

Clock Movies. This dataset is based on the film The Clock (2010), which is a 24-hour film montage of different movies featuring clocks. We collect 1,244 images from over 600 different movies from the movie's Fandom page. As the timestamp within the movie reflects the absolute time by design, the time label can be implicitly obtained.

5.6.2 Implementation details

For the localisation stage, we use an off-the-shelf detector CBNetV2 [Liang et al. 2021] to crop the clocks, and add 20% context to each side of the image to ensure recall. The detector is trained on COCO [T.-Y. Lin et al. 2014] and achieves the state-of-the-art performance on COCO among methods with publicly released models at the time of writing.

We first train the recognition model on SynClock dataset. During training, the dataset is generated on-the-fly. We train the model using the Adam [Kingma and Ba 2015] optimiser with learning rate 1e-4 and batch size 32 for 100k iterations.

We then use the model trained on SynClock to generate pseudo-labels for the Clock Time-lapse dataset, and filter valid ones into the training set using uniformity constraints. We then fine-tune the trained model on the enlarged training set for 20k iterations with batch size 64, with half being from SynClock and half being from the pseudo-labelled dataset. We then repeat the process where we use the newly trained model to generate pseudo-labels, and then retrain the model using the same settings as the first retraining.

5.6.3 Evaluation metrics

As this is a new task, there is no existing method for evaluation. We therefore propose the following metrics:

- **Hour accuracy.** when the predicted hour is correct.
- **Minute accuracy.** when the predicted minute is correct within a +-1 margin. We think this is close to the limit of human readability in resolving the in-between cases, especially with the absence of the second hand.
- **Overall accuracy.** when both hour and minute are correct (minute within a +-1 margin).

We also look into top-1, 2, 3 prediction accuracies, as some clocks have ambiguities with hands.

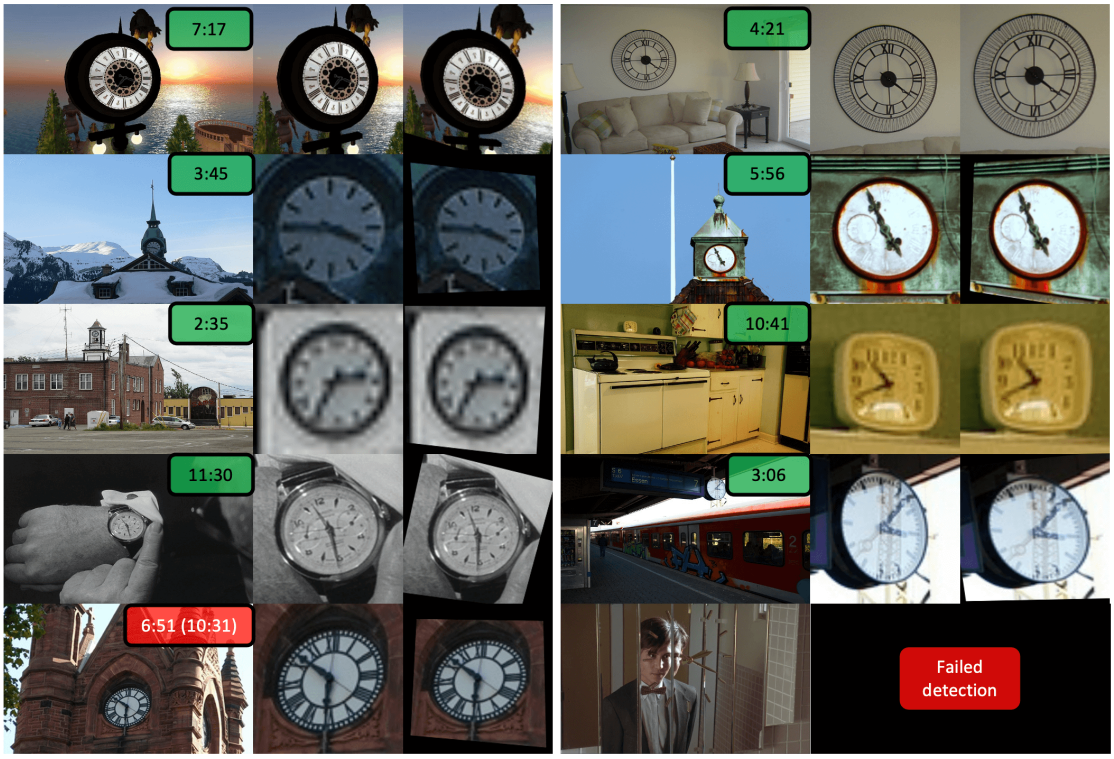


Figure 5.6: **Qualitative results.** The columns show the original, cropped, and canonical images, with predicted time overlaid on the original image. The model is able to read clocks in varying and challenging scenes, including those with different styles (rows 1-2), low resolution (row 3), and non-frontal viewing angles (row 4). The bottom row shows two failure cases: reading swapped hands and failed detection.

5.7 Results

Dataset	1	1-H	1-M	2	3
COCO	80.4	86.9	84.4	84.3	86.5
OpenImages	77.3	83.5	81.9	81.5	84.6
Clock Movies	79.0	82.3	82.4	83.3	85.5

Table 5.3: **End-to-end results.** We report the accuracy on the three datasets. 1, 2, 3 are the top 1, 2, and 3 overall accuracies respectively, whereas 1-H and 1-M are the top-1 hour and minute accuracies. The model achieves success in all these datasets.

5.7.1 End-to-end results

For a successful detection, both localisation and recognition have to be successful. To our best knowledge, no existing work has been able to achieve success in this setting, so it is challenging to compare to prior work. The results for the full model are shown in Table 5.3. We show that the model is able to achieve success in all these challenging datasets. Note that the accuracy is lower than that reported in

Dataset	mAP	AP50	AP75
COCO-val	77.0	92.1	74.6
OpenImages	61.5	89.2	66.4

Table 5.4: **Localisation results.** We report the average precision (AP) of the bounding box with respect to the ground truth. Note that the object detector itself is trained on COCO-train, so only its validation set is reported.

	COCO (IOU>50)					OpenImages (IOU>50)				
Training	1	1-H	1-M	2	3	1	1-H	1-M	2	3
SynClock	12.2	25.9	27.1	16.5	19.4	13.8	28.8	26.2	17.5	20.9
+ Augmentation	16.3	30.7	29.8	21.5	25.2	19.0	33.8	30.9	23.2	25.9
+ Homography	47.8	58.4	58.6	57.1	61.9	44.2	54.6	56.5	51.5	56.5
+ Artefacts	57.9	68.2	65.0	66.2	70.7	53.2	63.2	61.5	58.4	63.3
+ Spatial Transformer	59.6	71.3	67.0	67.1	71.4	55.4	65.8	64.2	61.8	65.9
+ Timelapse Round 1	73.9	81.5	79.1	78.6	81.7	69.5	76.7	75.6	74.1	77.4
+ Round 2	80.2	86.3	84.6	83.9	86.5	75.8	81.7	81.3	80.1	83.1
+ Round 3	82.6	88.1	85.7	86.2	88.1	79.0	83.5	83.5	81.7	84.5
+ Round 4	82.8	88.4	86.1	86.4	88.6	78.9	83.6	83.5	82.4	85.5
+ Round 5	82.9	88.4	86.8	86.6	88.5	79.6	84.7	83.9	83.3	86.2

Table 5.5: **Recognition results**. Here, all evaluations are conducted on clocks of IOU>50 with respect to the groundtruth annotations, to isolate the effect of detection failure and focus solely on incorrect predictions from recognition failure. We show that augmentation, alignment (STN), and pseudo-labelling are all essential components for improving the performance of recognition module.

the last row of Table 5.5, as this also includes the cases where the detection is unsuccessful.

5.7.2 Localisation-only results

To disentangle the effects of localisation and recognition, we firstly report the results for localisation on COCO and OpenImages in Table 5.4. To be consistent with the object detection literature, we report the average precisions (AP50, AP75), where the bounding box IoU is over a threshold. We also report the mean average precision (mAP), which is the average of APs across the thresholds [50:5:95]. Overall, the detection task is reasonably successful.

5.7.3 Recognition-only results

To evaluate the performance only on recognition, we only select images where the bounding box IoU is above 50%. Table 5.5 shows incrementally the effects of dif-

ferent components within the model.

SynClock. We show the effects of parts that constitute SynClock, namely data augmentation, homography transformation, and artefacts. The homography contributes the most towards accuracy (+31.5%/+25.2% for COCO/OpenImages top-1 accuracy respectively) as it allows clock reading from various angles and sizes. Augmentations (+4.1%/+5.2%) such as blurring and jittering and artefacts (+10.1%/+9.0%) such as shadows and random lines both help bridge the Syn2Real generalisation gap.

Spatial transformer. We show the effect of the spatial transformer within the architecture, which results in a monotonic improvement across all metrics (+1.7%/+2.2%).

Pseudo-labelling videos. We show that adding pseudo-labelled real video to the training set greatly contributes towards accuracy (+14.3%/+14.1%). Iteratively repeating the process also yields further improvements (+9.0%/+10.1%).

5.7.4 Qualitative visualisation

We show the qualitative results in Figure 5.6, from localisation, alignment and recognition. Each process can potentially give errors, which will impact the performance. We show that our model generalises to various styles of clocks, and is able to overcome low-resolution and alignment problems. The failure case is also shown in the bottom row.

The case where the model reads swapped hands is a good example of the limitation of our model. In this example, the hands appear to be of similar length. We (humans) can resolve this ambiguity and tell the correct time (10:31) because we look at the relative position between the hour hand and the hour mark. That is, if the time were to be 6:51 (as the model incorrectly predicted), then the hour hand should be closer to 7, and not 6. The model is not yet able to reason to this level, and hence makes wrong predictions.

5.8 Conclusion

This work introduces a framework for clock reading in real images or videos. We also circumvent the lack of training data in the recognition stage by proposing the synthetic dataset generator SynClock, and iteratively pseudo-labelling on real unlabelled videos using uniformity constraints. Further, we propose three benchmark datasets with accurate time labels. In the future, clock reading should become a standard processing step for images, in the same way that text spotting and object detection are now.

Acknowledgements

We thank Yimeng Long and Emily Tsen for assistance on data annotation, Joao Carreira for an interesting discussion, and Guanqi Zhan, Ragav Sachdeva, K R Prajwal, and Aleksandar Shtedritski for proofreading. This research is supported by the UK EPSRC CDT in AIMS (EP/S024050/1), and the UK EPSRC Programme Grant Visual AI (EP/T028572/1).

Appendices

The appendices can be found on the online version of the paper.

Statement of authorship

A statement of authorship for this paper is provided in the Appendix.

Chapter 6

Made to Order: Discovering monotonic temporal changes via self-supervised video ordering

The paper was published at the European Conference on Computer Vision, 2024 as an Oral Presentation.

timeless

Generated with Meta AI

Made to Order: Discovering monotonic temporal changes via self-supervised video ordering

Charig Yang[1] Weidi Xie[1,2] Andrew Zisserman[1]

[1]VGG, Department of Engineering Science, University of Oxford

[2]Shanghai Jiao Tong University

{charig,weidi,az}@robots.ox.ac.uk

https://www.charigyang.github.io/order

Abstract

Our objective is to discover and localize monotonic temporal changes in a sequence of images. To achieve this, we exploit a simple proxy task of ordering a shuffled image sequence, with 'time' serving as a supervisory signal, since only changes that are monotonic with time can give rise to the correct ordering. We also introduce a transformer-based model for ordering of image sequences of arbitrary length with built-in attribution maps. After training, the model successfully discovers and localizes monotonic changes while ignoring cyclic and stochastic ones. We demonstrate applications of the model in multiple domains covering different scene and object types, discovering both object-level and environmental changes in unseen sequences. We also demonstrate that the attention-based attribution maps function as effective prompts for segmenting the changing regions, and that the learned representations can be used for downstream applications. Finally, we show that the model achieves the state-of-the-art on standard benchmarks for image ordering.

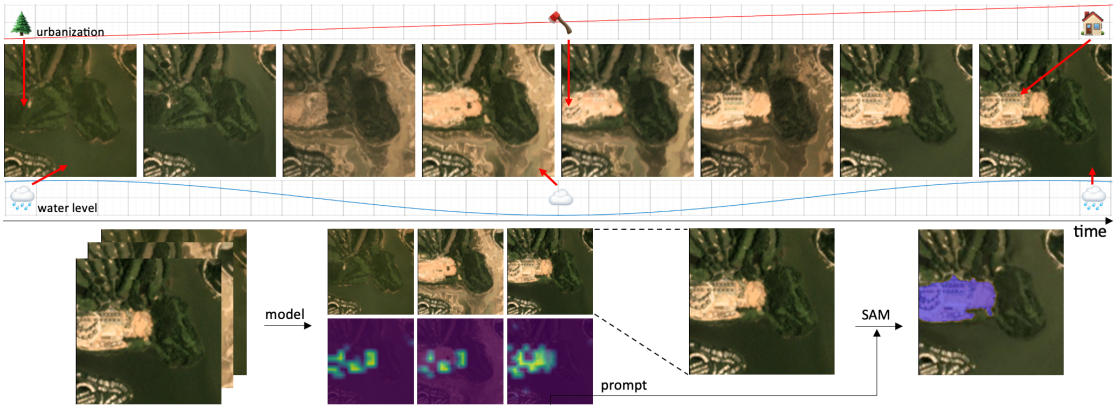


Figure 6.1: **Localizing monotonic temporal changes.** Top: satellite images (ordered left to right) taken months apart, containing several changes – some are monotonic (e.g. urbanization), while others are seasonal/cyclic (e.g. water level). Bottom: Our model's attribution map prediction on the sequence is able to localize the regions with monotonic temporal changes (in green), while being invariant to the seasonal and sporadic changes. The model is trained with no manual supervision, generalises to unseen sequences (as here), and the attribution map can also be used as a prompt to obtain segmentation.

6.1 Introduction

In the image sequence in Figure 6.1, there exists numerous changes over time, though many of these are seasonal, and hence distracting for long-term monitoring applications. As humans, we not only observe what is changing, but also reason about which changes are correlated monotonically with time and which ones are not. In this paper, we introduce a new task of automatically identifying temporally correlated changes in an image sequence, while being invariant to other changes. More specifically we wish to *discover* and *localize* the image regions where the change is correlated with time. Our motivation is to go beyond just detecting changes, but also discovering what changes are relevant over a period of time, and to explore the potential applications that this task could enable.

To achieve this, we propose a self-supervised proxy task, with time serving as a supervisory signal: the task is simply to order a shuffled sequence of video frames. The insight is that **frames are only orderable if changes are monotonic**, therefore if a model can order the video frames, it would have learned to identify relevant (monotonic temporal-correlated) cues while disregarding other changes. The trained model can then be employed for video analysis applications where the goal is to identify **changes** over time, such as developments or deforestation in

satellite imagery (whilst ignoring seasonal variations) (see Figure 6.1) or aging signs in medical images. It can also be employed for detecting and tracking monotonic object **motion** over time, such as shadows caused by the movement of the sun or animals moving smoothly across the scene.

It is worth noting that temporal ordering has previously been used as a proxy task for self-supervised representation learning, with the learnt representations then finetuned for downstream tasks, such as video action recognition [D. Wei et al. 2018; Misra et al. 2016; H.-Y. Lee et al. 2017; Fernando et al. 2017] (though due to the disparity between the proxy and downstream tasks, the effectiveness of learnt visual representation from temporal ordering has been unsatisfactory compared to other proxy tasks [T. Chen et al. 2021; T. Han et al. 2020; Grill et al. 2021; He et al. 2022]). In contrast, the objective in this paper is to use ordering as a proxy task to directly train a model for discovering and localizing monotonic changes in video sequences, without any subsequent supervised finetuning. In this sense, we are similar in spirit to previous works that use self-supervision to directly solve tasks, such as [Jabri et al. 2020; Lai et al. 2020; Bian et al. 2022; C. Yang et al. 2021] that targets tracking and segmentation by training on proxy tasks.

In order to harness the ordering proxy task, we introduce a transformer-based model that is able (i) to perform *ordering* on natural images sequences, and, more importantly, (ii) to provide *attribution* by localizing the evidence that gives rise to its prediction. Specifically, we use a DETR-like transformer encoder-decoder architecture where the queries in the decoder are cast as an *ordering index*. The architecture is designed to allow an attribution map to be obtained directly as part of the proxy training. Once the model has been trained for a particular setting, such as change detection in satellite image sequences, then it can generalize to unseen videos in the same setting, requiring only a forward pass to predict the localization of the monotonic temporal changes from the attribution map.

To summarize, in this paper, we make the following four contributions: (i) we introduce a new task of discovering and localizing monotonic temporal changes in image sequences, and establish the link that temporal ordering can be used as a self-supervised proxy task for training; (ii) we introduce a flexible transformer-based architecture for ordering that can also automatically localize monotonic changes; (iii) once trained on a setting (such as satellite images), the model is

able to discover the changes correlating monotonically with time in unseen image sequences in the same setting, and we demonstrate several situations where this can be applied. Finally, (iv) we demonstrate that the trained model is able to order novel image sequences surpassing the performance of previous approaches for ordering on standard benchmarks.

6.2 Related work

Video self-supervised learning has become increasingly popular in computer vision. Most research in this area focuses on representation learning, with an emphasis on downstream performance after supervised fine-tuning or linear probing. In contrast, there is a less explored direction that goes beyond representation learning to directly learning a useful task under the self-supervised learning setting, such as depth [T. Zhou et al. 2017; Godard et al. 2019], optical flow [Meister et al. 2018; P. Liu et al. 2019], correspondence [X. Wang et al. 2019] and sound localization [Afouras et al. 2020; Arandjelovic and Zisserman 2018; H. Chen et al. 2021; J. Liu et al. 2022]. Our work in this paper builds upon such paradigm, that enables to deploy the model to downstream tasks without the need for labels.

Self-supervised learning from time. This work involves using temporal ordering as a supervisory signal. Alternative sources of supervision is to leverage other cues, such as speed [Benaim et al. 2020], uniformity [C. Yang et al. 2022], and the arrow of time [D. Wei et al. 2018]. The closest kin to our work involves using temporal sequencing as supervision [Misra et al. 2016; Fernando et al. 2017; H.-Y. Lee et al. 2017; Fernando et al. 2015], though their primary focus is on representation learning. In this work, we focus on advocating temporal ordering as a useful task on its own, showing that localization can emerge by using time as supervisory signal. We highlight the differences between our work and others in the Supplementary Material.

Ordering has been a longstanding task in computer science, dating back to sorting algorithms. In machine learning, it is a relevant task in both language [X. Chen et al. 2016; Cui et al. 2018] and vision [Misra et al. 2016; Basha et al. 2012; Zhukov et al. 2020; Shvetsova et al. 2023]. For images, ordering has also been treated as a pretext for self-supervised pretraining, such as jigsaw puzzles [Noroozi and Favaro

2016], or as a task of interest, for example, image sequencing [Basha et al. 2013; Basha et al. 2012; Zarrabi et al. 2018; Sevilla-Lara et al. 2021; H. Kim and Sabuncu 2023]. There is also some interest in the literature that focuses on differentiable sorting algorithms [Petersen et al. 2021; Petersen et al. 2022; Cuturi et al. 2019; Grover et al. 2019; A. Brown et al. 2020], though they mostly focus on algorithmic developments, such as differentiable loss function and black-box combinatorial optimisation. While in this paper, we make contributions towards the architecture by introducing a transformer-based ordering model, which allows ordering of arbitrary-length image sequences, with built-in visualisation via attribution maps.

Attribution localization, specifically in the case where explicit supervision is not given, has been of interest in the vision community. In ConvNets, attribution methods attempt to look into the network to find out where it is seeing [B. Zhou et al. 2016; Fong and Vedaldi 2017]. In transformers, several methods have been proposed to look into the attention on the `[CLS]` tokens [Abnar and Zuidema 2020]. Instead of these implicit localization, several methods have also carefully designed the architecture so that localization emerges explicitly despite not being trained on, such as in sound localization [Afouras et al. 2020; Arandjelovic and Zisserman 2018; H. Chen et al. 2021; J. Liu et al. 2022]. This work follows the latter paradigm, while using the self-supervision from ordering.

Change detection has also been studied in computer vision. Many works look at changes in the image domain [Sachdeva and Zisserman 2023; Sakurada and Okatani 2015], and across different applications from construction monitoring [Stent et al. 2015], satellite imaging [Mall et al. 2023], to medical imaging [Patriarche and Erickson 2004]. Other works associate short-term changes with motion, and use motion as a cue to discover moving objects [Bideau and Learned-Miller 2016b; Bideau and Learned-Miller 2016a; Lamdouar et al. 2020; C. Yang et al. 2021; Lamdouar et al. 2021; J. Xie et al. 2022]. We differ from these lines of work in that we are mostly concerned with temporally coherent changes at different timescales, which may go beyond object level and not associated with motion.

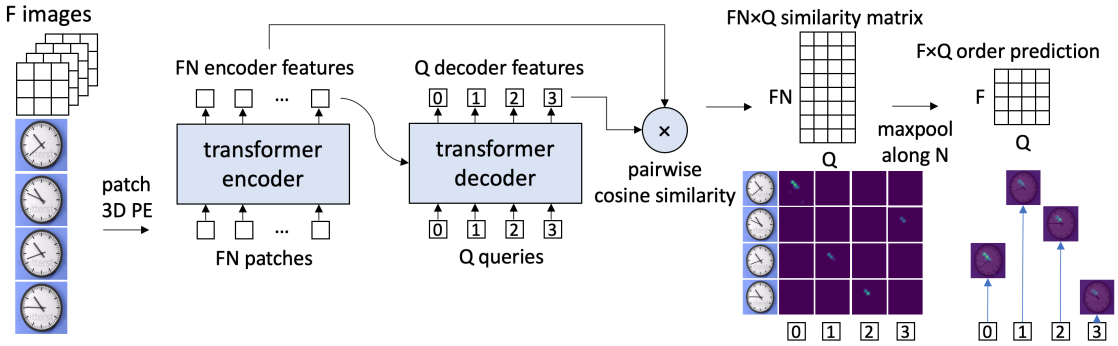


Figure 6.2: **Network architecture.** For an unordered sequence of F frames each with N patches, the transformer encoder takes in all FN patches as input, and outputs FN features. The transformer decoder takes in Q learnable queries, each corresponding to an ordinal position, and the encoder output for cross-attention, resulting in Q features for output. A $FN \times Q$ cosine similarity matrix is constructed between all pairs of features from the encoder and decoder outputs, and the spatial max-pooling over this matrix reveals the $F \times Q$ order predictions. The ordering can simply then be obtained by taking an argmax along each query axis. In the example sequence, the hour hand is correlated monotonically with time, and appears in the attribution map.

6.3 Method

6.3.1 Problem formulation

Our goal is to train a vision model to localize the changes in an image sequence that correlate monotonically with time. As a subsidiary goal, the model should also be able to order the image sequence.

Formally, given set of images, the model should output an *attribution map* $\mathbf{S}_{\text{att}} \in \mathbb{R}^{F \times H \times W}$ and an ordering $y_{\text{order}} \in \mathbb{Z}^F : y_{\text{order},i} \in \{0, 1, 2, ..., F-1\}$ as:

$$y_{\text{order}}, \mathbf{S}_{\text{att}} = \Phi(\mathcal{I}_0, \mathcal{I}_1, ..., \mathcal{I}_{F-1})$$

where $\mathcal{I} \in \mathbb{R}^{C \times H \times W}$ represents the input images. We show that we can train the model Φ via self-supervised learning on a proxy task, namely, ordering an arbitrary sequence of F images shuffled from a temporal sequence.

6.3.2 Ordering architecture

To address this problem, we propose a simple yet novel transformer-based architecture, as shown in Figure 6.2. The architecture comprises a transformer encoder (Φ_{enc}) that encodes the image patches, and a transformer decoder (Φ_{dec})

that encodes the ordering. To obtain an attribution map, we simply compute the pairwise cosine similarity between features from the encoder and queries from the decoder. We can then perform a max pooling operation across patches of the same image to get the ordering prediction.

Transformer Encoder (Φ_{enc}). To process an unordered sequence of F images, *i.e.*, $\mathcal{X} = \{\mathcal{I}_0, \mathcal{I}_1, ..., \mathcal{I}_{F-1}\}$, we start by dividing each frame $\mathcal{I} \in \mathbb{R}^{C \times H \times W}$ into 2D patches of size (P, P), resulting in $N = HW/P^2$ patches per frame and FN patches in total. Following the vision transformer approach, we flatten each patch using a learnable projection layer to D dimensions and add 3D positional encoding (spatial and frame) to each patch.

It is important to note that the frame positional encoding does not contain absolute temporal information since the frames are unordered, but it allows the patches to identify whether they belong to the same frame. As a result, after patchifying the input sequence, it ends up with a tensor of $\mathbf{x} \in \mathbb{R}^{F \times N \times D}$, which is then fed into a transformer encoder. The key difference to the standard vision transformer is that we output all the features, *i.e.*, $\mathbf{x}_{\text{enc}} \in \mathbb{R}^{F \times N \times D}$ instead of using a `[CLS]` token. In summary, we can express this procedure as $\mathbf{x}_{\text{enc}} = \Phi_{\text{enc}}(\mathcal{I}_0, \mathcal{I}_1, ..., \mathcal{I}_{F-1})$.

Transformer Decoder (Φ_{dec}). The transformer decoder is composed of Q learnable queries $\mathbf{q} \in \mathbb{R}^{Q \times D}$, with each corresponding to an ordering position $(0, 1, ..., Q-1)$. The task for the transformer decoder is to align the query vector with the encoder feature that demonstrates the correct temporal order. These queries iteratively attend the visual outputs from the encoder with cross-attention in the standard transformer decoder. We denote the output of the decoder as, $\mathbf{x}_{\text{dec}} \in \mathbb{R}^{Q \times D} = \Phi_{\text{dec}}(\mathbf{q}, \mathbf{x}_{\text{enc}})$. In practice, $Q = F$.

Cosine similarity matrix (S). Recall that we now possess two sets of features: encoder features $\mathbf{x}_{\text{enc}} \in \mathbb{R}^{F \times N \times D}$ and decoder features $\mathbf{x}_{\text{dec}} \in \mathbb{R}^{Q \times D}$. We then compute the pairwise cosine similarity matrix $\mathbf{S} \in \mathbb{R}^{F \times N \times Q} : [\mathbf{S}]_{i,j} = \cos(\mathbf{x}_{\text{enc},i}, \mathbf{x}_{\text{dec},j}) \in [-1, 1]$ between each i of the $F \times N$ features in $\mathbf{x}_{\text{enc}}$ and each j of the Q features in $\mathbf{x}_{\text{dec}}$, where $\cos(\cdot, \cdot)$ denotes the cosine similarity function.

Given the similarity matrix, we want to obtain (i) the ordering of the frames and (ii) the attribution map that indicates the spatial evidence within each frame that gives rise to ordering. The matrix $\mathbf{S} \in \mathbb{R}^{F \times N \times Q}$ consists of $F \times Q$ different spatial

maps of size N, each indicating the attention between each pair of queries ($j \in Q$) and images ($i \in F$).

Order prediction. To obtain the order predictions, we perform spatial max-pooling over the patches of each frame (along the N dimension), to obtain $\hat{y} \in \mathbb{R}^{F \times Q} = \max_{i \in N} \mathbf{S}_i$. This max-pooling is designed to create an information bottleneck – the query has to attend to the correct token(s) within the correct image in order to predict the order correctly. The resulting matrix serves as a predictor for the position of each query in the ordering. We then apply a softmax along the query axis of the matrix to get the probability scores for each query.

Attribution map. Among the $F \times Q$ different spatial maps, we are only interested in the ones that correspond to the correct ordering. For each query $j \in Q$, we select one map $i \in F$ that has the maximum activation, *i.e.* $i = \arg\max \hat{y}_j$ resulting in Q maps. We then rearrange and resize each map of N patches back to the original resolution, resulting in $\mathbf{S}_{\text{att}} \in \mathbb{R}^{Q \times H \times W}$. Notably, this localization can be achieved without the need for additional fine-tuning, supervision, or post-hoc attribution methods [Fong and Vedaldi 2017; Abnar and Zuidema 2020].

6.3.3 Training and inference

Temporal loss. Given the ground-truth order $y \in \mathbb{Z}^Q : y_i \in \{0, 1, ..., F-1\}$, the model can be trained via binary cross-entropy loss. Specifically, we convert the ground-truth order into a binary permutation matrix. With some notation abuse, we still call this matrix $y \in \mathbb{Z}^{F \times Q}$. The forward loss is then simply the elementwise binary cross-entropy between the two matrices: $\mathcal{L}_f(y, \hat{y}) = \frac{1}{FQ} \sum_{i \in F} \sum_{j \in Q} \text{cross-entropy}(\hat{y}_{ij}, y_{ij})$.

In practice, we find that allowing reversibility in the loss aids with training, as many changes are reversible in nature without prior knowledge of the arrow of time [D. Wei et al. 2018] (the sequence could equally be ordered from first to last, or last to first). To allow this, we calculate the loss as the minimum of the loss for both forward and backward sequences, *i.e.* $\mathcal{L}_r = \min(\mathcal{L}_f(y, \hat{y}), \mathcal{L}_f(\text{reverse}(y), \hat{y}))$. This loss is zero when the model predicts the order correctly in either direction.

Inference. At inference time, we simply take the argmax along each query axis as the order prediction, that is, $y_{\text{order}} \in \mathbb{Z}^Q : y_{\text{order},j} = \arg\max_{i \in F} \hat{y}_{i,j}$ In other

words, each query picks the image that contains the maximum activation for its query, as illustrated in the bottom-right corner of Figure 6.2.

6.3.4 Discussion

Generalization to different sequence lengths. Our architecture is designed to handle sequences of arbitrary, possibly unequal length during training and inference, without the need to re-design or train separate models for each sequence length. At training time, we assume there is a maximum number of images, thus initialize a total of F_{max} learnable queries in the transformer, *i.e.*, $\mathbf{q} \in \mathbb{R}^{F_{\text{max}} \times D}$. While the model handles a sequence of F images, with $F \leq F_{max}$, it only uses the first F queries as input to the decoder, ignoring the rest. This approach enables each query to represent its positioning $(0, 1, ..., F-1)$, making it generalizable to different lengths during both training and testing. However, the model will not generalize to lengths above F_{max}.

Avoiding trivial solutions. There are two factors that we need to account for: camera motion and video compression artifacts. Camera motion can be smooth or uniform over a short time gap, which can result in an uninteresting cue. To address this, we apply a small random cropping on each frame in settings where the time gap between frames is small (*i.e.* $<$ 1s). This slight jittering helps to prevent the model from learning trivial solutions. We note that this does not degrade the performance even if camera motion is absent. Another factor that can give rise to trivial solutions is inter-frame video compression artifacts. To address this, we follow conventional wisdom [Iashin et al. 2022; D. Wei et al. 2018] and use H.264 formatting for all videos, thus minimizing compression artifacts and preventing trivial solutions.

From localization to segmentation. While the attribution map is useful, some applications may benefit from going beyond just localization. Here, we propose three solutions. (i) We can directly obtain segmentation at patch-level granularity by thresholding the attribution map, together with minimal post-processing namely averaging across frames and removing small contours. (ii) We replace the linear projection layer on image patches with a pretrained DINOv2 to enhance the feature quality. (iii) Alternatively, we can use the highest activation points (*i.e.* centre pixels of patches) as prompt for the pretrined Segment Anything model

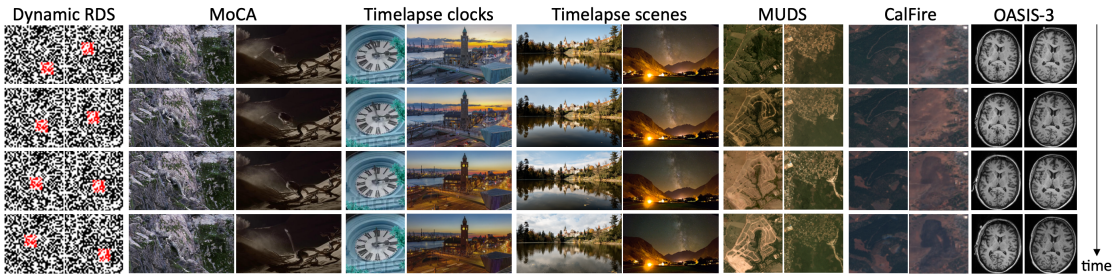


Figure 6.3: **Sequence datasets.** From left to right: dynamic Random Dot Stereograms (RDS) (moving dots colored only for illustration), moving camouflaged animals (MoCA), timelapse clocks (cropped/full), timelapse scenes, MUDS, CalFire, OASIS-3.

(SAM) [Kirillov et al. 2023], to obtain pixel-level segmentation masks.

6.4 Experiments

6.4.1 Video datasets

To leverage the inherent temporal information in videos, we sample a sequence from a video, shuffle the frames, then train the model with the original ordering as groundtruth. This enables the attribution map to identify monotonic changes that contribute to the ordering while disregarding other changes. Our study explores sequences across multiple domains, each with distinct cues of interest, which we summarize in Table 6.1 and illustrate sample sequences in Figure 6.3.

Dynamic random dot stereograms are a type of image sequence that features a box of random dots moving smoothly over a background of random dots, originally used as a pair to demonstrate stereoscopic motion [Neff et al. 1985]. While the individual images may appear random, the box is visible when viewed in sequence. We generate a synthetic dataset of controllable dynamic random dot stereograms (Dynamic RDS) to test the model's ability to detect subtle relative cues. Since this a synthetic dataset, we know the ground-truth of the box's motion, so can compare this with the predictions.

Moving camouflaged animals (MoCA) [Lamdouar et al. 2020] was constructed from videos of camouflaged animals. We use this dataset to investigate the use of short-term object locomotion as a cue, particularly where it is challenging to distinguish the object from the background. To accomplish this, we follow [C. Yang et al. 2021] and focus on the subset of 88 videos in which the

animals are in motion. To evaluate on localisation, we assume that the change is object-level due to animal motion, and use the annotated object bounding box as the ground-truth for localization.

Timelapse analog clocks. Our study examines real-world scenes that feature both absolute cues (time on clocks) and relative cues (scene changes). To accomplish this, we utilize the Timelapse dataset [C. Yang et al. 2022] in two ways. Firstly, we use the entire dataset of 2,511 videos that features cropped clocks. Secondly, we create a subset consisting of 260 outdoor scenes with static cameras where the clock occupies only a small area of the scene.

Timelapse scenes. We gathered outdoor timelapse videos from WebVid-10M dataset [Bain et al. 2021] with 'timelapse' as the search query. The dataset comprises 180 static videos, and our aim is to investigate the cues that the model can extract from the scene to learn its order, as there are no specific absolute cues present.

6.4.2 Temporal image sequences

We also show effectiveness of our approach for image sequences that are collected over a period of time, including satellite images and longitudinal medical data.

Multi-temporal urban development dataset (MUDS) [Van Etten et al. 2021] is a dataset that contains 80 aerial satellite image sequences captured monthly over a two-year period. We aim to identify geographical changes that occur over time, including deforestation and urbanization, while ignoring other changes. As there are no existing datasets and benchmarks for this task, we hand-label 60 subsequences on the test set for segmentation masks where changes are monotonic, and call this evaluation set **Monotonic MUDS**. Samples are shown in Figure 6.4a. We use this dataset to evaluate localization and segmentation performance of the model trained on the MUDS dataset.

CalFire [Mall et al. 2022] is a satellite dataset tracking wildfires in California. It also contains other events, such as snowfall, new construction, changes in water level, that we aim to discover. We pre-process by removing scenes with significant cloud cover.

OASIS-3 [LaMontagne et al. 2019] contains longitudinal MRI scans taking 1-4

dataset	Δt	cue	seq (trn/test)	EM	EW
Dynamic RDS	<1s	motion	∞	99.8	99.9
MoCA	<1s	motion	75/13	82.0	90.6
Clocks (cropped)	∼1m	clock	2011/500	62.5	73.0
Clocks (full)	∼1h	clock/scene	210/20	55.0	74.3
Timelapse scenes	∼1h	scene	130/50	61.8	79.4
MUDS	1mo	landscape	60/20	56.4	69.6
CalFire	1mo	landscape	800/276	76.6	87.5
OASIS-3	1y	brain	100/34	84.3	89.1

Table 6.1: **Dataset attributes and ordering results.** This table shows different datasets and their attributes, as well as the ordering results on the test (unseen) sequences on exact match (EM) and elementwise (EW) metrics.

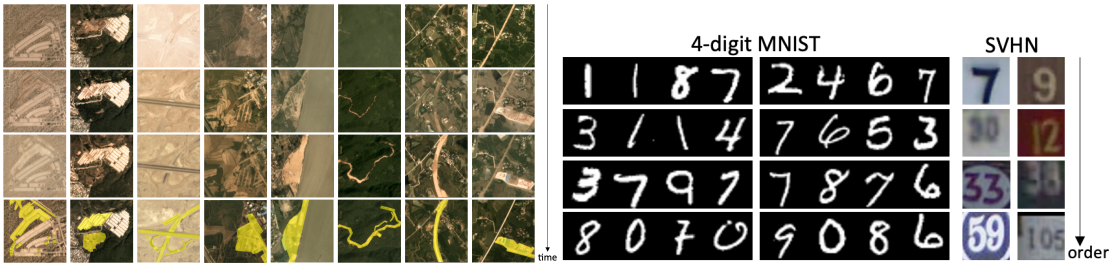
years apart to study how brains age. We select sequences with 3 or more scans, resulting in 134 sequences. Following [H. Kim and Sabuncu 2023], we perform affine registration and use the centre slice.

6.4.3 Image ordering datasets

In addition, we showcase our general ordering capability by evaluating our method on standard benchmarks. We compare our ordering performance with related works [Petersen et al. 2021; Petersen et al. 2022; Grover et al. 2019; Cuturi et al. 2019] on the task of sorting images of numbers in ascending order, as shown in Figure 6.4b. What the model has to learn here is different from the previous datasets, as the ordering is *absolute*, and not understanding changes.

Multi-digit MNIST [LeCun et al. 1998] dataset is a modified version of the MNIST dataset in which four digits are concatenated to form a four-digit number. The goal is to order the image sequences in increasing order. To construct this dataset, we synthetically combine examples from the corresponding train and test sets of the MNIST dataset, resulting in a total of 50,000 training and 10,000 testing images.

Street view house numbers (SVHN) [Netzer et al. 2011], was collected from Google Street View and includes images of house numbers. Similarly, the task is to order these numbers in increasing order. The dataset consists of 33,402 images for training and 13,068 for testing. To ensure consistency with previous studies, we followed the data preprocessing and augmentation methods described in [Goodfellow et al. 2013].



(a) **Monotonic MUDS benchmark.** (b) **Image ordering datasets.**

Figure 6.4: (a) To evaluate localization and segmentation performance, we manually annotate the monotonically changing regions (shown in yellow) on the MUDS test set. Each sequence contains four frames, and the monotonic changes between the first and last frames are annotated. (b) 4-digit MNIST [LeCun et al. 1998] (left) and SVHN [Netzer et al. 2011] (right). The task is to order the images by the numbers they contain in increasing order (top to bottom).

6.4.4 Evaluation metrics

Localization. We use a pointing-game evaluation method, this is to follow the convention of the localization literature in other domains, including audio-visual localization [Afouras et al. 2020; Arandjelovic and Zisserman 2018] and saliency methods [Fong and Vedaldi 2017; Fong et al. 2019], that is, if the pixel with maximum activation in the attribution map contains the change of interest, then it is positive (1), otherwise it is negative (0). As our method only outputs patch-level attribution, we simply select the centre pixel of the patch as the highest activation. The overall accuracy is then calculated as the average over all sequences.

Segmentation. We use the standard metric for segmentation, *i.e.*, mean intersection over union (mIoU), where the mean is the average across all sequences.

Ordering. We use the evaluation metrics for sorting as outlined in [Petersen et al. 2021; Petersen et al. 2022]. These metrics include exact match (EM) and elementwise (EW) accuracy. EM is considered correct if the entire sequence is ordered correctly, while EW considers the order accuracy per element. Following previous benchmarks we evaluate these at sequence lengths 5, 9 and 16. To test the generalization to different sequence lengths, we also evaluate the exact match accuracy at a fixed sequence length of 5 at test-time (EM5), regardless of the training sequence length.

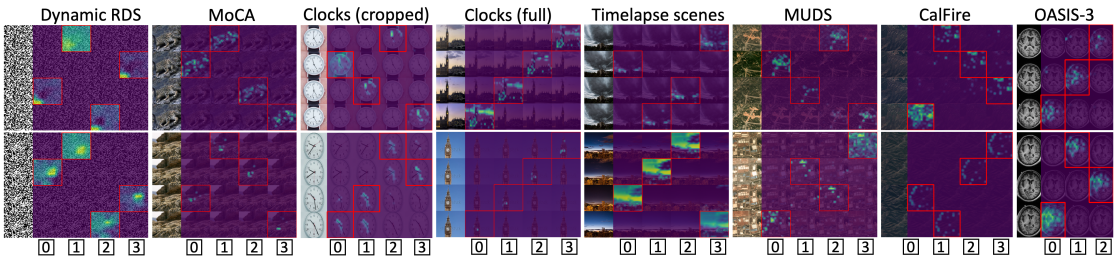


Figure 6.5: **Ordering and localization results** across various datasets, where the model is able to discover and localize various cues across different domains, including object motion, clocks, scenery, landscape and biological aging. The left column shows the input (unordered) images. Each column of the similarity matrix represents the model's prediction of each individual order (0, 1, 2, 3), where the image in the red box is chosen, and the attention heat map within the box localizes the change.

6.4.5 Implementation details

We split each dataset into disjoint training and testing subsets, and then randomly sample frames from each video. We keep the time gap between sampled frames constant within each video, but vary this across videos to train a robust model. For image sequences (MUDS, CalFire, OASIS-3) where data collection is less regular, we relax these constraints and simply randomly sample between all frames within the sequence. We train each model separately for each dataset.

Architecture. For the encoder, we use a smaller version of TimeSFormer [Bertasius et al. 2021] with a divided space-time attention architecture, consisting of 256 dimensions, 4 heads, 6 layers, and 512 MLP dimensions. For the decoder, we use a standard transformer decoder with the same parameters, except for 64 dimensions and 3 layers. As a result, the model is lightweight, with only 4M parameters. We use Adam optimizer [Kingma and Ba 2015] with learning rate 1e-4 in all experiments, and batch size 32 sequences with 4 frames per sequence for all video datasets, except 3 frames for OASIS-3 as this is the minimum MRI scan sessions per subject. For image ordering, we use batch size 100 with varying numbers of frames. All experiments are run on a single GPU. The code, datasets, and models will be released.

6.5 Results

6.5.1 Results on ordering video frames or image sequence

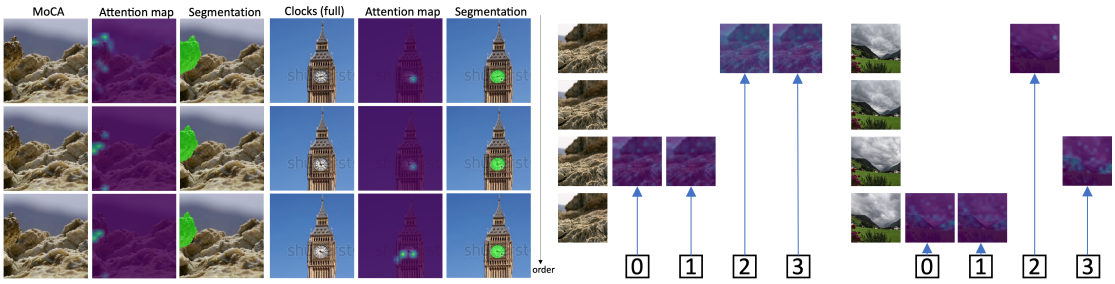
Video ordering results. The results for ordering as well as the main signals that the model can pick up, are shown by EM and EW in Table 6.1, with qualitative results in Figure 6.5. The model is able to successfully order sequences across different datasets, particularly in cases where changes are significant. It is expected that the scores are not perfect, as sampled data from videos is not guaranteed to contain ordering cues, as illustrated in the Supplementary.

Temporal image sequence ordering results. For satellite image sequences, the model is able to discover cues that are relevant to ordering, including road building and forest fires. This illustrates our model's application on remote sensing imagery. In MRI scans, we explore the cues for age changes. Our results show that there are cues in the posterior part of the brain. This is consistent with the literature [Blinkouskaya and Weickenmeier 2021] that suggest that ventricular enlargement is a prominent feature, and causes the posterior horn to inflate in response to tissue loss. There are also some cues along the outline. This is in line with the literature [Svennerholm et al. 1997; Scahill et al. 2003], which suggest that brain volume also decreases with old age.

Failure cases. A limitation of our model is that we do not force a one-to-one matching between queries and images, and this may result in some images being claimed by multiple different queries or by none at all, as seen in Figure 6.6b. This problem can easily be resolved by allowing each image to be predicted once. However, not being able to order also provides valuable information as not all real sequences can be ordered – for example, sequences where everything is static for a period, or very stochastic. Therefore, we treat invalid orderings as a means to provide information on whether particular frames can be ordered or not. We report further experimental analyses on the failure cases in the Supplementary.

6.5.2 Comparison with change detection methods

We compare against three previous approaches for the ability to detect monotonic temporal changes on Monotonic MUDS dataset, namely, a supervised Siamese



(a) **SAM Prompting.** (b) **Failure cases.**

Figure 6.6: (a) The attribution map is used to prompt SAM to obtain segmentation masks. (b) Unorderable sequences, one being too static and the other being too stochastic.

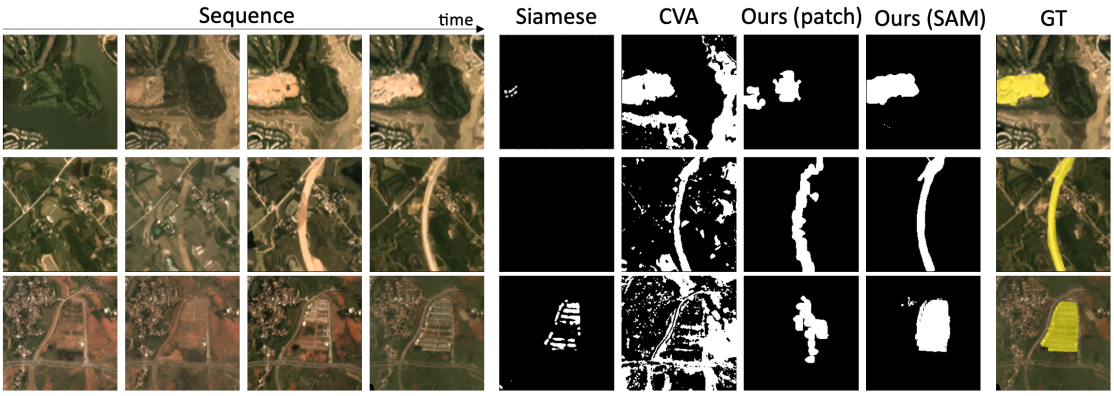


Figure 6.7: **Segmentation comparison** with other methods: Siamese networks [Hafner et al. 2022] and CVA [Malila 1980]. Supervised methods ignore changes other than building, and pixel-based methods over-segments non-monotonic regions.

Networks [Hafner et al. 2022] trained on MUDS dataset for the task of *urban development tracking*, by highlighting the differences between buildings; Change Vector Analysis [Malila 1980], which is a baseline for the task of *change detection*, highlighting all changes in an image pair without knowledge their nature, and Deep CVA [Saha et al. 2019], a learned version of CVA that has been trained specifically on images.

The performance comparison is given in Table 6.2, and illustrated in Figure 6.7. As can be seen, the urban development tracking method [Hafner et al. 2022] under-segments changes that are correlated with urbanisation, as it is only trained to look at buildings. The change detection methods of [Malila 1980; Saha et al. 2019] over-segments changes that are non-monotonic as it has no concept of time. Both prior methods have shortcomings in detecting urban development: the former method misses changes like road building and deforestation, and the latter includes many erroneous regions such as the seasonal changes in vegetation and water level. Our

Method	Mono-MUDS loc (acc) ↑	Mono-MUDS seg (mIoU) ↑	RDS seg (mIoU) ↑	MoCA loc (acc) ↑
Siamese [Hafner et al. 2022]	73.3	11.1	–	–
CVA [Malila 1980]	71.7	34.6	22.3	69.6
DCVA [Saha et al. 2019]	70.0	35.5	–	–
Ours	**83.3**	37.9	**34.2**	**75.0**
Ours + DINOv2	80.0	41.3	–	–
Ours + SAM	**83.3**	**45.1**	–	–

Table 6.2: **Localization and segmentation results** via the pointing game accuracy for localization, and mIoU for segmentation. The methods for segmentation (patch and SAM) are described in Section 6.3.4.

methods	loc (acc) ↑	seg (mIoU) ↑	finetuning (F1) ↑
scratch	—	—	18.2
S&L [Misra et al. 2016]	23.3	20.9	15.8
OPN [H.-Y. Lee et al. 2017]	25.0	24.1	17.1
Ours (AoT proxy)	63.3	26.9	25.5
Ours	**83.3**	**45.1**	**30.2**

Table 6.3: **Comparison with self-supervised methods** on localization and segmentation on Mono-MUDS and on fine-tuning for building change detection on MUDS.

model is able to highlight correctly the monotonic changes while being invariant to other changes. We further note that our model *discovers* such changes without any prior information on what to look for. Additionally, replacing the patch projection with DINOv2 and using the localization to prompt SAM both improve the results. We also evaluate on two other datasets: RDS and MoCA, where we have ground-truth for the moving objects in the video; and show that we obtain favourable results. Note that Siamese and DCVA are only trained on satellite images, hence do not generalize to other domains.

Quantitatively, in Table 6.2, we find that (i) our pointing-game localization and patch-level segmentation results outperform other methods despite operating only on patch-level (7×7) and not pixel-level granularity, and (ii) segmentation via SAM prompting further improves the results.

Qualitatively, the results of our localization experiments on various video datasets are presented in Figure 6.5. The model is capable of accurately identifying monotonic changes while remaining invariant to unrelated cues. Notably, these results were achieved on sequences unseen during training. We additionally show that our localization map works as a good prompt for the Segment Anything (SAM) model to obtain object-level changes, as in Figure 6.6a.

6.5.3 Comparison on self-supervised proxy tasks

Here, we compare to other self-supervised methods based on *time*. We include a discussion in the Supplementary on the subtle differences between the proxy tasks. We include results for training baselines from scratch on MUDS, and testing on Mono-MUDS in localizing and segmenting monotonic changes. The results are shown in Table 6.3, where we conduct three sets of experiments, as detailed below.

First, we observe that previous methods are extremely crude in localization; this is expected, as they are all based on conv5 feature (even AoT, via CAM). Given a 224p input, conv5 has a 13×13 feature map with 195p receptive field. Our architecture handles localization by design, and is hence more capable than other methods that use post-hoc attribution methods on top of standard backbones. We also note that AoT does not train, which is also expected as it ingests optical flow as input (and in satellite images using flow does not make sense (change≠motion)), our method is more flexible in this regard.

Second, we ask if our method is still superior if the architectural gap is closed. To achieve this, we also compare the proxy task in AoT (time direction) with ours (ordering) under a fairer comparison (using our architecture and RGB input), and show this in the table under "Ours (AoT proxy)". We conclude that both our architecture and proxy task leads to significant improvements.

Third, we investigate fine-tuning on a target task that requires both time and localization: change detection of buildings (like [Hafner et al. 2022]). For each method, we pre-train the encoder on MUDS, freeze it, then train a lightweight head (3× deconvolutions) on top of the same dataset, and test on unseen sequences. To keep this fair, we keep the number of parameters roughly the same across methods. The results (Table 6.3, right column) show that our method learns good representations as compared to previous self-supervised methods in localization tasks.

6.5.4 Comparison with image ordering methods

We compare to two previous methods on image ordering benchmarks where the task is to arrange the images in increasing order. *Differentiable sorting networks* such as DiffSort [Petersen et al. 2021] and its successor DSortv2 [Petersen et al.

dataset	MNIST			SVHN		
frames	5	9	16	5	9	16
DSort [Petersen et al. 2021]	83.4|92.6	56.3|86.7	30.5|80.7|86.6	64.1|82.1	24.2|69.6	3.9|59.6|66.8
DSv2 [Petersen et al. 2022]	84.9|93.2	63.8|89.1	31.1|82.2|—	68.5|84.1	39.9|75.8	12.2|65.6|—
Ptr-Net [Vinyals et al. 2015]	91.9|95.6	87.7|95.0	68.9|90.0|1.1	76.3|87.6	48.7|79.4	9.8|63.2|0.1
Ours	**93.9**|**96.7**	**87.9**|**95.2**	**72.2**|**91.2**|**92.9**	**77.3**|**88.2**	**53.9**|**81.0**	**19.4**|**67.9**|**67.6**

Table 6.4: **Ordering on image datasets** on two standard benchmarks (MNIST and SVHN) where the task is to order images of numbers in increasing order. Metrics are (EM|EW|EM5). EM and EW are evaluated at the sequence length the model has been trained on (5/9/16), whereas EM5 tests generalisation to test length 5.

2022] employ a parameter-free sorting network to rank scalars. *Pointer Networks* [Vinyals et al. 2015] ranks features by using a recurrent encoder-decoder network with attention. We note that Ptr-Net is not initially designed for such a task, but for arranging a set of coordinates. We simply extend pointer networks by adding an image encoder and task the model to rank the image features from small to large. We then jointly train this encoder and the pointer network. For fair comparison, we use the same transformer encoder as our model, and use the pointer network as the decoder with similar size to our transformer decoder.

The quantitative results are presented in Table 6.4. Our results demonstrate that (i) we compare favourably in ordering performance – on both MNIST and SVHN, our method has the best performance of the four (ii) Ptr-Net does not automatically generalise to testing with different sequence lengths, while our method does (as reflected by the poor EM5 accuracy), and (iii) our method also has the added benefit of having an attribution map.

6.6 Conclusion

In this paper, we explore using time as a proxy loss for self-supervised training of models to discover and localize monotonic temporal changes in image sequences. Possible extensions include discovering more complex temporal changes (seasonal/periodic), or object state and attribute changes. It would also be interesting to investigate how the model scales with larger datasets and compute, and what new applications this task can enable. Overall, we hope this paper presents a valuable starting point for future research and applications in this area.

Acknowledgements

We thank Tengda Han, Ragav Sachdeva, and Aleksandar Shtedritski for suggestions and proofreading. This research is supported by the UK EPSRC CDT in AIMS (EP/S024050/1), and the UK EPSRC Programme Grant Visual AI (EP/T028572/1).

Appendices

The appendices can be found on the online version of the paper.

Statement of authorship

A statement of authorship for this paper is provided in the Appendix.

Chapter 7

Discussion

7.1 Achievements and impact

This section outlines the achievements and impact of the papers. Overall, this thesis presents some pioneering works in discovering objects using motion (Chapters 2, 3, 4), new applications previously unachievable in computer vision (Chapters 5, 6). We also present new datasets and benchmarks (Chapters 2, 5, 6) and contribute to the field in learning without manual annotations for different applications (Chapters 3, 5, 6).

At the time of writing, the published papers cumulatively received over 275 combined citations, and over 600 stars on GitHub. Excitingly, many of the papers have had impacts that went beyond what was anticipated at the time of publication.

7.1.1 Camouflaged object discovery

In Chapter 2, we presented a simple solution of breaking camouflage using motion, and the first large-scale dataset of moving camouflaged animals. To date, the paper has received 80 citations.

Video camouflaged object detection. Our work was one of the pioneering works for video camouflaged object detection (VCOD) in the deep learning era, and the simple model based on registration and memory has been improved to handle these difficult scenarios. For example, [X. Cheng et al. 2022] proposes to handle motion implicitly instead of relying on optical flow, [Lamdouar et al.

2021] proposes a dual-head architecture, and [Meunier et al. 2022] proposes an expectation-maximisation framework.

MoCA dataset. The dataset is, to date, the largest dataset for video camouflaged object discovery, and has been used to evaluate motion segmentation models, including those presented in Chapters 3, 4 and 6. [X. Cheng et al. 2022] extended our bounding-box annotations to include segmentation masks, which has become a popular benchmark for VCOD. [P. Zhang et al. 2024; P. Zhang et al. 2023] also used the subset of our dataset featuring underwater animals to study fish segmentation.

7.1.2 Motion grouping

In Chapter 3, we extended this idea towards discovering objects in general videos (video object segmentation), and show that a simple model using only the motion stream without manual annotations can achieve high accuracy. This paper has received 170 citations, and the abbreviated version of this paper received the Best Paper Award at the Robust Video Scene Understanding workshop at CVPR 2021.

A (re)focus on motion. Although motion segmentation has been studied for a long time [D. Sun et al. 2012], the quality has not been satisfactory compared to RGB counterparts, partly due to the low quality of optical flow predictions. Our contributions were particularly timely, as advancements in motion estimation [Teed and Deng 2020] have significantly improved optical flow quality, enabling motion segmentation to achieve considerable success on general videos without relying on appearance. Our work hence influenced the community's interest towards using motion in a more meaningful way. Follow-up works include extending our method of clustering motion directly [J. Xie et al. 2022; Lamdouar et al. 2021], or using motion as a training signal for RGB-based segmentation [Elsayed et al. 2022; Choudhury et al. 2022].

Applications. The method for segmentation via motion grouping has led to several applications beyond the initial expectations. Our work has been used to improve structure-from-motion [Goli et al. 2024], in scene decomposition for generating sprites [Ye et al. 2022] or 3D avatars [Guo et al. 2023], and in decomposing videos into layers [Shrivastava et al. 2024]. Also, [S. Han et al. 2024] adapts our

method of slot attention on motion to understand biological motion perception.

7.1.3 Motion segmentation using SAM

In Chapter 4, we presented a simple method of using SAM to perform motion segmentation by treating it as a direct segmentation task. This paper was selected for Oral Presentation at ACCV 2024. It has also generated traction on social media, with over 150k video views on $\mathbb{X}$[1].

An off-the-shelf motion segmentation model. This paper received 285 stars on GitHub, the highest among all of the papers in this thesis. Thanks to their simplicity and strength, the models in this paper have been used as an off-the-shelf motion segmentation model that can be applied across domains. In "MotionStone: Decoupled Motion Intensity Modulation with Diffusion Transformer for Image-to-Video Generation", [S. Shi et al. 2024] used it to create a motion mask for video generation, while in "Reanimating Images using Neural Representations of Dynamic Stimuli", [Yeung et al. 2024] used the model in the pipeline to visualise the dynamic stimuli within the human brain.

Advancing the progress of motion segmentation. Motion segmentation has been one of the main topics of focus on this thesis, and all three of the papers have contributed to advancing the frontier in this area over the years, as evident by the increasingly accurate segmentation in each of the papers. The method presented in this chapter represents the current state-of-the-art.

7.1.4 Analog clock reading

In Chapter 5, we introduced the paper that solves a niche yet overlooked task of analog clock reading, and providing both training and benchmarking datasets. This paper received the Best Presentation Award at the UK Robotics CDT Conference in 2022.

Training and benchmarking vision-language models. The recent rise of general-purpose large vision-language models highlighted their inability to perform simple yet specialised tasks, one of which being reading analog clocks. The datasets contributed in the paper were used to both train [Q. Sun et al. 2024] and evaluate

[1] https://x.com/dreamingtulpa/status/1782083341883224462

[Deitke et al. 2024] vision-language models for this specialised task, highlighting how explorations can lead to surprising outcomes.

Public outreach. The reach of the work goes beyond the computer vision research community. Being a task that many non-researchers can relate to, the work has generated much organic reviews including from prominent figures in the field, and has been published in the New Scientist[2], a magazine with 143k weekly circulation and 4.2M monthly website visits.

7.1.5 Monotonic change discovery

In Chapter 6, we presented a method for detecting monotonic changes using ordering as self-supervision. This paper was selected for Oral Presentation at ECCV 2024. While this is a new paper, it has already started to gain impact. Particularly, in "EarthDial: Turning Multi-sensory Earth Observations to Interactive Dialogues", [Soni et al. 2024] used our dataset to both train and evaluate a vision-language model based on multi-temporal earth observations.

7.2 Future work

Unifying appearance and motion across time. Part I of the thesis deals with motion segmentation using optical flow as an intermediate representation by disentangling motion completely. This comes with several limitations (i) the accuracy of segmentation is limited by the (imperfect) optical flow, (ii) it is difficult to model long-term temporal changes, (iii) it involves redundant computation as motion is implicit within video frames, (iv) it is unclear how best to re-introduce appearance information into the model, (v) it is unclear when motion is useful and what to do when motion is not.

Ideally, a model should be able to take in a series of video frames as input, and output consistent segmentation masks for each frame as output. The model can rely on appearance in case motion is not necessary (still objects), and rely on motion in case where visibility is limited (camouflaged or transparent objects). One potential solution is to combine the two models proposed in Chapter 4, that take both appearance and motion, and use SAM2 [Ravi et al. 2024] for setting up

[2] https://www.newscientist.com/article/2298773

memory across frames.

Understanding the physical world through temporal reasoning. Part II of the thesis deals with using the property of time, specifically uniformity and monotonicity, to understand videos through temporal relationships. However, these constraints are rather strong, and real-world changes can be seasonal and noisy, limiting the applicability of the method. A reasonable extension in this direction is to also detect different nature of changes including cyclic change, such as seasonal ones, and also be more robust to small stochasticities while still capturing the global trends.

This way, a method can be developed to perform a "Fourier Transform" for videos, decomposing them into a combination of different time-varying components. This will allow videos to be treated analogous to time series, which will allow more advanced analyses especially over long periods of time.

A natural step after learning temporal correlations is to move towards understanding causality. The current model learns from scratch from the data itself without any prior knowledge. One possible solution is to leverage vision-language models in order to perform reasoning about the nature of changes over time.

Deployment, adoption, and scale. This thesis introduces several dramatically simplified methods for achieving high performance on existing tasks (Chapters 2-4), as well as new applications of computer vision that have never been shown to be possible in the literature (Chapters 5-6).

While this is a sufficient reward in itself, a natural next step is to see how these applications can be deployed in practical scenarios. For example, the system developed for detecting camouflages in Chapters 2-4 will be of interest for researchers working in large-scale monitoring systems (such as wildlife or industry), where our model will be able to 'flag' moving things that may be of interest. Similarly, the model for detecting monotonic changes in Chapter 6 can potentially help with longitudinal monitoring over long periods of time, both in remote sensing and medical imaging.

Also, none of the models in this thesis has been trained on more than a single GPU at a time. Scaling this up both in terms of compute and data will allow better understanding of the models.

7.3 Conclusion

Reflecting upon the concept of time has always been a fascinating exploration across physics, psychology and philosophy. While the ability to manipulate time remains in fantasies, the growing accessibility of photography and digital processing enables us to capture moments frozen in time. This thesis draws parallels from our understanding of time, and contributes towards making machines understand the temporal dimension of the world by learning from time (to time).

References

Abnar, S. and Zuidema, W. (2020). "Quantifying attention flow in transformers". In: *arXiv preprint arXiv:2005.00928*.

Afouras, T., Owens, A., Chung, J. S., and Zisserman, A. (2020). "Self-supervised learning of audio-visual objects from video". In: *Proc. ECCV*.

Alayrac, J.-B., Carreira, J., Arandjelovic, R., and Zisserman, A. (2019a). "Controllable Attention for Structured Layered Video Decomposition". In: *Proc. ICCV*.

Alayrac, J.-B., Carreira, J., and Zisserman, A. (2019b). "The Visual Centrifuge: Model-Free Layered Video Representations". In: *Proc. CVPR*.

Alexeev, A., Kukharev, G., Matveev, Y., and Matveev, A. (2020). "A highly efficient neural network solution for automated detection of pointer meters with different analog scales operating in different conditions". In: *Mathematics*.

Arandjelovic, R. and Zisserman, A. (2018). "Objects that sound". In: *Proc. ECCV*.

Bain, M., Nagrani, A., Varol, G., and Zisserman, A. (2021). "Frozen in Time: A Joint Video and Image Encoder for End-to-End Retrieval". In: *Proc. ICCV*.

Baker, S., Roth, S., Scharstein, D., Black, M. J., Lewis, J., and Szeliski, R. (2007). "A Database and Evaluation Methodology for Optical Flow". In: *Proc. ICCV*.

Ballas, N., Yao, L., Pal, C., and Courville, A. (2016). "Delving deeper into convolutional networks for learning video representations". In: *Proc. ICLR*.

Bao, H., Tan, Q., Liu, S., and Miao, J. (2019). "Computer Vision Measurement of Pointer Meter Readings Based on Inverse Perspective Mapping". In: *Applied Sciences*. URL: https://www.mdpi.com/2076-3417/9/18/3729.

Basha, T. D., Moses, Y., and Avidan, S. (2012). "Photo sequencing". In: *Proc. ECCV*.

Basha, T. D., Moses, Y., and Avidan, S. (2013). "Space-Time Tradeoffs in Photo Sequencing". In: *Proc. ICCV*.

Al-Baz, A. (2020). *Reading Analog Clocks with Xception CNN Network*. https://github.com/albazahm/Reading_Analog_Clocks_with_Xception_CNN_Network.

Benaim, S., Ephrat, A., Lang, O., Mosseri, I., Freeman, W. T., Rubinstein, M., Irani, M., and Dekel, T. (2020). "Speednet: Learning the speediness in videos". In: *Proc. CVPR*.

Bertasius, G., Wang, H., and Torresani, L. (2021). "Is Space-Time Attention All You Need for Video Understanding?" In: *Proc. ICML*.

Bian, Z., Jabri, A., Efros, A. A., and Owens, A. (2022). "Learning Pixel Trajectories with Multiscale Contrastive Random Walks". In: *Proc. CVPR*.

Bideau, P. and Learned-Miller, E. (2016a). "A detailed rubric for motion segmentation". In: *arXiv preprint arXiv:1610.10033*.

Bideau, P. and Learned-Miller, E. (2016b). "It's moving! A probabilistic model for causal motion segmentation in moving camera videos". In: *Proc. ECCV*.

Bideau, P., Menon, R. R., and Learned-Miller, E. (2018). "MoA-Net: self-supervised motion segmentation". In: *ECCV Workshop*.

Bilen, H., Fernando, B., Gavves, E., Vedaldi, A., and Gould, S. (2016). "Dynamic Image Networks for Action Recognition". In: *Proc. CVPR*.

Blinkouskaya, Y. and Weickenmeier, J. (2021). "Brain shape changes associated with cerebral atrophy in healthy aging and Alzheimer's disease". In: *Frontiers in Mechanical Engineering*.

Boltzmann, L. (1896). *Lectures on Gas Theory*. University of California Press. URL: https://books.google.co.uk/books?id=jUobO7s879EC.

Brachmann, E., Krull, A., Nowozin, S., Shotton, J., Michel, F., Gumhold, S., and Rother, C. (2017). "DSAC-differentiable RANSAC for camera localization". In: *Proc. CVPR*.

Brachmann, E. and Rother, C. (2018). "Learning less is more-6d camera localization via 3d surface regression". In: *Proc. CVPR*.

Brachmann, E. and Rother, C. (2019). "Neural- Guided RANSAC: Learning Where to Sample Model Hypotheses". In: *Proc. ICCV*.

Brostow, G. J. and Essa, I. A. (1999). "Motion based decompositing of video". In: *Proc. ICCV*.

Brown, A., Xie, W., Kalogeiton, V., and Zisserman, A. (2020). "Smooth-AP: Smoothing the Path Towards Large-Scale Image Retrieval". In: *Proc. ECCV*.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B.,

Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). "Language Models are Few-Shot Learners". In: *NeurIPS*.

Brox, T. and Malik, J. (2010). "Object segmentation by long term analysis of point trajectories". In: *Proc. ECCV*.

Burgess, C. P., Matthey, L., Watters, N., Kabra, R., Higgins, I., Botvinick, M., and Lerchner, A. (2019). "MONet: Unsupervised Scene Decomposition and Representation". In: *arXiv preprint arXiv:1901.11390*.

Butler, D. J., Wulff, J., Stanley, G. B., and Black, M. J. (2012). "A naturalistic open source movie for optical flow evaluation". In: *Proc. ECCV*.

Caelles, S., Maninis, K.-K., Pont-Tuset, J., Leal-Taixé, L., Cremers, D., and Van Gool, L. (2017). "One-shot video object segmentation". In: *Proc. CVPR*.

Cai, W., Ma, B., Zhang, L., and Han, Y. (2020). "A pointer meter recognition method based on virtual sample generation technology". In: *Measurements*.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. (2020). "End-to-end object detection with transformers". In: *Proc. ECCV*.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. (2021). "Emerging Properties in Self-Supervised Vision Transformers". In: *Proc. ICCV*.

Cen, J., Fang, J., Yang, C., Xie, L., Zhang, X., Shen, W., and Tian, Q. (2023). "Segment Any 3D Gaussians". In: *arXiv preprint arXiv:2312.00860*.

Chen, H., Xie, W., Afouras, T., Nagrani, A., Vedaldi, A., and Zisserman, A. (2021). "Localizing Visual Sounds the Hard Way". In: *Proc. CVPR*.

Chen, M., Radford, A., Child, R., Wu, J., Jun, H., Luan, D., and Sutskever, I. (2020). "Generative Pretraining From Pixels". In: *Proc. ICML*.

Chen, T., Zhu, L., Ding, C., Cao, R., Zhang, S., Wang, Y., Li, Z., Sun, L., Mao, P., and Zang, Y. (2023). "SAM-Adapter: Adapting Segment Anything in Underperformed Scenes". In: *ICCV Workshop*.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2021). "A Simple Framework for Contrastive Learning of Visual Representations". In: *Proc. ICML*.

Chen, X., Qiu, X., and Huang, X. (2016). "Neural sentence ordering". In: *arXiv preprint arXiv:1607.06952*.

Cheng, H. K., Oh, S. W., Price, B., Schwing, A., and Lee, J.-Y. (2023). "Tracking Anything with Decoupled Video Segmentation". In: *Proc. ICCV*.

Cheng, H. K. and Schwing, A. G. (2022). "XMem: Long-Term Video Object Segmentation with an Atkinson-Shiffrin Memory Model". In: *Proc. ECCV*.

Cheng, J., Tsai, Y.-H., Wang, S., and Yang, M.-H. (2017). “Segflow: Joint learning for video object segmentation and optical flow”. In: *Proc. ICCV*.

Cheng, X., Xiong, H., Fan, D.-P., Zhong, Y., Harandi, M., Drummond, T., and Ge, Z. (2022). “Implicit motion handling for video camouflaged object detection”. In: *Proc. CVPR*.

Cheng, Y., Li, L., Xu, Y., Li, X., Yang, Z., Wang, W., and Yang, Y. (2023). “Segment and track anything”. In: *arXiv preprint arXiv:2305.06558*.

Cho, D., Hong, S., Kang, S., and Kim, J. (2019). “Key Instance Selection for Unsupervised Video Object Segmentation”. In: *arXiv preprint arXiv:1906.07851*.

Cho, S., Lee, M., Lee, S., Park, C., Kim, D., and Lee, S. (2023). “Treating Motion as Option to Reduce Motion Dependency in Unsupervised Video Object Segmentation”. In: *Proc. WACV*.

Choudhury, S., Karazija, L., Laina, I., Vedaldi, A., and Rupprecht, C. (2022). “Guess What Moves: Unsupervised Video and Image Segmentation by Anticipating Motion”. In: *Proc. BMVC*.

Cui, B., Li, Y., Chen, M., and Zhang, Z. (2018). “Deep attentive sentence ordering network”. In: *EMNLP*.

Cuturi, M., Teboul, O., and Vert, J.-P. (2019). “Differentiable ranking and sorting using optimal transport”. In: *NeurIPS*.

Dave, A., Tokmakov, P., and Ramanan, D. (2019). “Towards segmenting anything that moves”. In: *ICCV Workshop*.

Deitke, M., Clark, C., Lee, S., Tripathi, R., Yang, Y., Park, J. S., Salehi, M., Muennighoff, N., Lo, K., Soldaini, L., Lu, J., Anderson, T., Bransom, E., Ehsani, K., Ngo, H., Chen, Y., Patel, A., Yatskar, M., Callison-Burch, C., Head, A., Hendrix, R., Bastani, F., VanderBilt, E., Lambert, N., Chou, Y., Chheda, A., Sparks, J., Skjonsberg, S., Schmitz, M., Sarnat, A., Bischoff, B., Walsh, P., Newell, C., Wolters, P., Gupta, T., Zeng, K.-H., Borchardt, J., Groeneveld, D., Nam, C., Lebrecht, S., Wittlif, C., Schoenick, C., Michel, O., Krishna, R., Weihs, L., Smith, N. A., Hajishirzi, H., Girshick, R., Farhadi, A., and Kembhavi, A. (2024). “Molmo and pixmo: Open weights and open data for state-of-the-art multimodal models”. In: *arXiv preprint arXiv:2409.17146*.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Li, F.-F. (2009). “ImageNet: A Large-Scale Hierarchical Image Database”. In: *Proc. CVPR*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *arXiv preprint arXiv:1810.04805*.

Doersch, C., Gupta, A., Markeeva, L., Recasens, A., Smaira, L., Aytar, Y., Carreira, J., Zisserman, A., and Yang, Y. (2022). "Tap-vid: A benchmark for tracking any point in a video". In: *NeurIPS*.

Doersch, C. and Zisserman, A. (2019). "Sim2real transfer learning for 3D human pose estimation: motion to the rescue". In: *NeurIPS*.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). "An image is worth 16x16 words: Transformers for image recognition at scale". In: *Proc. ICLR*.

Duvallet, F. (2016). *Deep time reading*. https://github.com/felixduvallet/deep-time-reading.

Dwibedi, D., Aytar, Y., Tompson, J., Sermanet, P., and Zisserman, A. (2020). "Counting out time: Class agnostic video repetition counting in the wild". In: *Proc. CVPR*.

Einstein, A. (1922). "The general theory of relativity". In: *The meaning of relativity*. Springer.

Elsayed, G., Mahendran, A., Van Steenkiste, S., Greff, K., Mozer, M. C., and Kipf, T. (2022). "Savi++: Towards end-to-end object-centric learning from real-world videos". In: *NeurIPS*.

Engelcke, M., Kosiorek, A. R., Jones, O. P., and Posner, I. (2020). "GENESIS: Generative scene inference and sampling with object-centric latent representations". In: *Proc. ICLR*.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. (2010). "The PASCAL Visual Object Classes (VOC) Challenge". In: *IJCV*.

Faktor, A. and Irani, M. (2014). "Video Segmentation by Non-Local Consensus voting." In: *Proc. BMVC*.

Fan, D.-P., Wang, W., Cheng, M.-M., and Shen, J. (2019). "Shifting more attention to video salient object detection". In: *Proc. CVPR*.

Fernando, B., Bilen, H., Gavves, E., and Gould, S. (2017). "Self-supervised video representation learning with odd-one-out networks". In: *Proc. CVPR*.

Fernando, B., Gavves, E., Oramas, J. M., Ghodrati, A., and Tuytelaars, T. (2015). "Modeling video evolution for action recognition". In: *Proc. CVPR*.

Fischler, M. A. and Bolles, R. C. (1981). "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM*.

Fong, R., Patrick, M., and Vedaldi, A. (2019). “Understanding deep networks via extremal perturbations and smooth masks”. In: *Proc. ICCV*.

Fong, R. and Vedaldi, A. (2017). “Interpretable explanations of black boxes by meaningful perturbation”. In: *Proc. ICCV*.

Fragkiadaki, K., Zhang, G., and Shi, J. (2012). “Video segmentation by tracing discontinuities in a trajectory embedding”. In: *Proc. CVPR*.

Gandelsman, Y., Shocher, A., and Irani, M. (2019). ““Double-DIP”: Unsupervised Image Decomposition via Coupled Deep-Image-Priors”. In: *Proc. CVPR*.

Girdhar, R., Carreira, J., Doersch, C., and Zisserman, A. (2019). “Video Action Transformer Network”. In: *Proc. CVPR*.

Godard, C., Mac Aodha, O., Firman, M., and Brostow, G. J. (2019). “Digging into self-supervised monocular depth estimation”. In: *Proc. ICCV*.

Goli, L., Sabour, S., Matthews, M., Brubaker, M., Lagun, D., Jacobson, A., Fleet, D. J., Saxena, S., and Tagliasacchi, A. (2024). “RoMo: Robust Motion Segmentation Improves Structure from Motion”. In: *arXiv preprint arXiv:2411.18650*.

Goodale, M. A. and Milner, A. D. (1992). “Separate visual pathways for perception and action”. In: *Trends in neurosciences*.

Goodfellow, I. J., Bulatov, Y., Ibarz, J., Arnoud, S., and Shet, V. (2013). “Multi-digit number recognition from street view imagery using deep convolutional neural networks”. In: *arXiv preprint arXiv:1312.6082*.

Gopnik, A. M., Meltzoff, A. N., and Kuhl, P. K. (2009). *The Scientist in the Crib: What Early Learning Tells Us About the Mind*. HarperCollins.

Greff, K., Kaufman, R. L., Kabra, R., Watters, N., Burgess, C., Zoran, D., Matthey, L., Botvinick, M., and Lerchner, A. (2019). “Multi-object representation learning with iterative variational inference”. In: *Proc. ICML*.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. A., Guo, Z. D., Azar, M. G., Piot, B., Kavukcuoglu, K., Munos, R., and Valko, M. (2021). “Bootstrap Your Own Latent: A New Approach to Self-Supervised Learning”. In: *NeurIPS*.

Grover, A., Wang, E., Zweig, A., and Ermon, S. (2019). “Stochastic optimization of sorting networks via continuous relaxations”. In: *Proc. ICLR*.

Guo, C., Jiang, T., Chen, X., Song, J., and Hilliges, O. (2023). “Vid2avatar: 3d avatar reconstruction from videos in the wild via self-supervised scene decomposition”. In: *Proc. CVPR*.

Gupta, A., Vedaldi, A., and Zisserman, A. (2016). "Synthetic data for text localisation in natural images". In: *Proc. CVPR*.

Hafner, S., Ban, Y., and Nascetti, A. (2022). "Urban change detection using a dual-task Siamese network and semi-supervised learning". In: *IGARSS*.

Han, S., Wang, Z., and Zhang, M. (2024). "Flow Snapshot Neurons in Action: Deep Neural Networks Generalize to Biological Motion Perception". In: *arXiv preprint arXiv:2405.16493*.

Han, T., Xie, W., and Zisserman, A. (2020). "Self-supervised Co-training for Video Representation Learning". In: *NeurIPS*.

Harley, A. W., Fang, Z., and Fragkiadaki, K. (2022). "Particle video revisited: Tracking through occlusions using point trajectories". In: *Proc. ECCV*.

Hartley, R. I. and Zisserman, A. (2000). *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521623049.

Hawking, S. (1998). *A Brief History of Time*. Bantam Books. URL: https://books.google.co.uk/books?id=E7mEnx3zE4AC.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. (2022). "Masked Autoencoders Are Scalable Vision Learners". In: *Proc. CVPR*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). "Deep residual learning for image recognition". In: *Proc. CVPR*.

Horn, B. K. and Schunck, B. G. (1981). "Determining optical flow". In: *Artificial intelligence*.

Horvath, B. (2021). *Analog clock reading*. https://github.com/aHorvathBazsi/Analog_clock_reading.

Hosler, B. C. and Stamm, M. C. (2020). "Detecting Video Speed Manipulation". In: *CVPR Workshop*.

Hossam, B. (2017). *clockreader*. https://github.com/basselhossam/clockreader.

Howells, B., Charles, J., and Cipolla, R. (2021). "Real-time analogue gauge transcription on mobile phone". In: *CVPR Workshop*.

Iashin, V., Xie, W., Rahtu, E., and Zisserman, A. (2022). "Sparse in Space and Time: Audio-visual Synchronisation with Trainable Selectors". In: *Proc. BMVC*.

Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., and Brox, T. (2017). "Flownet 2.0: Evolution of optical flow estimation with deep network". In: *Proc. CVPR*.

Jabri, A., Owens, A., and Efros, A. A. (2020). "Space-Time Correspondence as a Contrastive Random Walk". In: *NeurIPS*.

Jaderberg, M., Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). "Synthetic data and artificial neural networks for natural scene text recognition". In: *NeurIPS Workshop*.

Jaderberg, M., Simonyan, K., Zisserman, A., and Kavukcuoglu, K. (2015). "Spatial Transformer Networks". In: *NeurIPS*.

Jain, S. D., Xiong, B., and Grauman, K. (2017). "Fusionseg: Learning to combine motion and appearance for fully automatic segmentation of generic objects in videos". In: *Proc. CVPR*.

Johnson, J., Hariharan, B., van der Maaten, L., Fei-Fei, L., Zitnick, C. L., and Girshick, R. (2017). "CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning". In: *Proc. CVPR*.

Jojic, N. and Frey, B. J. (2001). "Learning flexible sprites in video layers". In: *Proc. CVPR*.

Jun Koh, Y. and Kim, C.-S. (2017). "Primary object segmentation in videos based on region augmentation and reduction". In: *Proc. CVPR*.

Karaev, N., Rocco, I., Graham, B., Neverova, N., Vedaldi, A., and Rupprecht, C. (2024). "Cotracker: It is better to track together". In: *Proc. ECCV*.

Keuper, M., Andres, B., and Brox, T. (2015). "Motion trajectory segmentation via minimum cost multicuts". In: *Proc. ICCV*.

Kim, H. and Sabuncu, M. R. (2023). "Learning to Compare Longitudinal Images". In: *MIDL*.

Kingma, D. P. and Ba, J. (2015). "Adam: A method for stochastic optimization". In: *Proc. ICLR*.

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., Dollár, P., and Girshick, R. (2023). "Segment Anything". In: *Proc. ICCV*.

Kumar, M. P., Torr, P. H., and Zisserman, A. (2008). "Learning layered motion segmentations of video". In: *IJCV*.

Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., Kamali, S., Popov, S., Malloci, M., Kolesnikov, A., Duerig, T., and Ferrari, V. (2020). "The Open Images Dataset V4: Unified image classification, object detection, and visual relationship detection at scale". In: *IJCV*.

Lai, Z., Lu, E., and Xie, W. (2020). "MAST: A Memory-Augmented Self-Supervised Tracker". In: *Proc. CVPR*.

Lai, Z. and Xie, W. (2019). "Self-supervised Learning for Video Correspondence Flow". In: *Proc. BMVC*.

Lamdouar, H., Xie, W., and Zisserman, A. (2021). “Segmenting Invisible Moving Objects”. In: *Proc. BMVC*.

Lamdouar, H., Yang, C., Xie, W., and Zisserman, A. (2020). “Betrayed by Motion: Camouflaged Object Discovery via Motion Segmentation”. In: *Proc. ACCV*.

LaMontagne, P. J., Benzinger, T. L., Morris, J. C., Keefe, S., Hornbeck, R., Xiong, C., Grant, E., Hassenstab, J., Moulder, K., Vlassenko, A. G., Raichle, M. E., Cruchaga, C., and Marcus, D. (2019). “OASIS-3: longitudinal neuroimaging, clinical, and cognitive dataset for normal aging and Alzheimer disease”. In: *MedRxiv*.

Laroca, R., Araujo, A. B., Zanlorensi, L. A., De Almeida, E. C., and Menotti, D. (2021). “Towards image-based automatic meter reading in unconstrained scenarios: A robust and efficient approach”. In: *IEEE Access*.

Le, T.-N., Nguyen, T. V., Nie, Z., Tran, M.-T., and Sugimoto, A. (2016). “Anabranch network for camouflaged object segmentation”. In: *CVIU*.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE*.

Lee, H.-Y., Huang, J.-B., Singh, M., and Yang, M.-H. (2017). “Unsupervised representation learning by sorting sequences”. In: *Proc. ICCV*.

Lee, M., Cho, S., Lee, S., Park, C., and Lee, S. (2023). “Unsupervised Video Object Segmentation via Prototype Memory Network”. In: *Proc. WACV*.

Li, F., Kim, T., Humayun, A., Tsai, D., and Rehg, J. M. (2013). “Video Segmentation by Tracking Many Figure-Ground Segments”. In: *Proc. ICCV*.

Li, S., Seybold, B., Vorobyov, A., Fathi, A., Huang, Q., and Kuo, C.-C. J. (2018). “Instance Embedding Transfer to Unsupervised Video Object Segmentation”. In: *Proc. CVPR*.

Liang, T., Chu, X., Liu, Y., Wang, Y., Tang, Z., Chu, W., Chen, J., and Ling, H. (2021). “CBNetV2: A Composite Backbone Network Architecture for Object Detection”. In: *arXiv preprint arXiv:2107.00420*.

Liao, M., Lyu, P., He, M., Yao, C., Wu, W., and Bai, X. (2021). “Mask TextSpotter: An End-to-End Trainable Neural Network for Spotting Text with Arbitrary Shapes”. In: *IEEE PAMI*.

Lin, H., Wu, R., Liu, S., Lu, J., and Jia, J. (2021). “Video Instance Segmentation with a Propose-Reduce Paradigm”. In: *Proc. ICCV*.

Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., and Dollar, P. (2014). “Microsoft COCO: Common Objects in Context.” In: *Proc. ECCV*.

Liu, J., Ju, C., Xie, W., and Zhang, Y. (2022). “Exploiting Transformation Invariance and Equivariance for Self-supervised Sound Localisation”. In: *ACM MM*.

Liu, L., Zhang, J., He, R., Liu, Y., Wang, Y., Tai, Y., Luo, D., Wang, C., Li, J., and Huang, F. (2020). “Learning by Analogy: Reliable Supervision from Transformations for Unsupervised Optical Flow Estimation”. In: *Proc. CVPR*.

Liu, P., Lyu, M., King, I., and Xu, J. (2019). “Selflow: Self-supervised learning of optical flow”. In: *Proc. CVPR*.

Liu, X., Liang, D., Yan, S., Chen, D., Qiao, Y., and Yan, J. (2018). “Fots: Fast oriented text spotting with a unified network”. In: *Proc. CVPR*.

Liu, Y., Chen, H., Shen, C., He, T., Jin, L., and Wang, L. (2020). “ABCNet: Real-time Scene Text Spotting with Adaptive Bezier-Curve Network”. In: *Proc. CVPR*.

Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., and Kipf, T. (2020). “Object-centric learning with slot attention”. In: *NeurIPS*.

Lowe, D. (1999). “Object recognition from local scale-invariant features”. In: *Proc. ICCV*.

Lu, E., Cole, F., Dekel, T., Xie, W., Zisserman, A., Salesin, D., Freeman, W. T., and Rubinstein, M. (2020). “Layered Neural Rendering for Retiming People in Video”. In: *SIGGRAPH Asia*.

Lu, X., Wang, W., Ma, C., Shen, J., Shao, L., and Porikli, F. (2019). “See More, Know More: Unsupervised Video Object Segmentation With Co-Attention Siamese Networks”. In: *Proc. CVPR*.

Lucas, B. D. and Kanade, T. (1981). “An iterative image registration technique with an application to stereo vision”. In: *IJCAI*.

Luiten, J., Zulfikar, I. E., and Leibe, B. (2020). “UnOVOST: Unsupervised Offline Video Object Segmentation and Tracking”. In: *Proc. WACV*.

Lyu, P., Liao, M., Yao, C., Wu, W., and Bai, X. (2018). “Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes”. In: *Proc. ECCV*.

Ma, J., He, Y., Li, F., Han, L., You, C., and Wang, B. (2024). “Segment anything in medical images”. In: *Nature Communications*.

Ma, L., Ye, Y., Hong, F., Guzov, V., Jiang, Y., Postyeni, R., Pesqueira, L., Gamino, A., Baiyya, V., Kim, H. J., Bailey, K., Fosas, D. S., Liu, C. K., Liu, Z., Engel, J., Nardi, R. D., and Newcombe, R. (2024). “Nymeria: A massive collection of multimodal egocentric daily motion in the wild”. In: *Proc. ECCV*.

Mahadevan, S., Athar, A., Ošep, A., Hennen, S., Leal-Taixé, L., and Leibe, B. (2020). "Making a Case for 3D Convolutions for Object Segmentation in Videos". In: *Proc. BMVC*.

Mahendran, A., Thewlis, J., and Vedaldi, A. (2018a). "Cross Pixel Optical-Flow Similarity for Self-Supervised Learning". In: *Proc. ACCV*.

Mahendran, A., Thewlis, J., and Vedaldi, A. (2018b). "Self-Supervised Segmentation by Grouping Optical-Flow". In: *Proc. ECCV*.

Malila, W. A. (1980). "Change vector analysis: An approach for detecting forest changes with Landsat". In: *LARS symposia*.

Mall, U., Hariharan, B., and Bala, K. (2022). "Change event dataset for discovery from spatio-temporal remote sensing imagery". In: *NeurIPS*.

Mall, U., Hariharan, B., and Bala, K. (2023). "Change-Aware Sampling and Contrastive Learning for Satellite Images". In: *Proc. CVPR*.

Maninis, K.-K., Caelles, S., Pont-Tuset, J., and Van Gool, L. (2018). "Deep extreme cut: From extreme points to object segmentation". In: *Proc. CVPR*.

Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., and Brox, T. (2016). "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation". In: *Proc. CVPR*.

Meister, S., Hur, J., and Roth, S. (2018). "Unflow: Unsupervised learning of optical flow with a bidirectional census loss". In: *AAAI*.

Meunier, E., Badoual, A., and Bouthemy, P. (2022). "EM-driven unsupervised learning for efficient motion segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Meunier, E. and Bouthemy, P. (2023). "Unsupervised Space-Time Network for Temporally-Consistent Segmentation of Multiple Motions". In: *Proc. CVPR*.

Miao, B., Bennamoun, M., Gao, Y., and Mian, A. (2022). "Self-Supervised Video Object Segmentation by Motion-Aware Mask Propagation". In.

Misra, I., Zitnick, C. L., and Hebert, M. (2016). "Shuffle and Learn: Unsupervised Learning using Temporal Order Verification". In: *Proc. ECCV*.

Moo Yi, K., Trulls, E., Ono, Y., Lepetit, V., Salzmann, M., and Fua, P. (2018). "Learning to find good correspondences". In: *Proc. CVPR*.

El-Naggar, A. O., Adam, H. M., El-Molla, Z. M., and El-shobokshy, H. (2018). *Convert from analog to digital clock*. https://github.com/Abdelrahman-Elnaggar/Convert-from-analog-to-digital-clock.

Naim, A., Aaroud, A., Akodadi, K., and El Hachimi, C. (2021). "A fully AI-based system to automate water meter data collection in Morocco country". In: *Array*.

Neff, R., Schwartz, S., and Stork, D. G. (1985). “Electronics for generating simultaneous random-dot cyclopean and monocular stimuli”. In: *Behavior Research Methods, Instruments, & Computers*.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. (2011). “Reading digits in natural images with unsupervised feature learning”. In: *NeurIPS*.

Noroozi, M. and Favaro, P. (2016). “Unsupervised learning of visual representations by solving jigsaw puzzles”. In: *Proc. ECCV*.

Ochs, P. and Brox, T. (2011). “Object segmentation in video: a hierarchical variational approach for turning point trajectories into dense regions”. In: *Proc. ICCV*.

Ochs, P., Malik, J., and Brox, T. (2014). “Segmentation of Moving Objects by Long Term Video Analysis”. In: *TPAMI*.

Oh, S. W., Lee, J.-Y., Xu, N., and Kim, S. J. (2019). “Video object segmentation using space-time memory networks”. In: *Proc. ICCV*.

Oh, T.-H., Jaroensri, R., Kim, C., Elgharib, M., Durand, F., Freeman, W. T., and Matusik, W. (Sept. 2018). “Learning-based Video Motion Magnification”. In: *Proc. ECCV*.

Palmer, S. E. (1999). *Vision Science: Photons to Phenomenology*. Bradford Book.

Pan, X., Li, P., Yang, Z., Zhou, H., Zhou, C., Yang, H., Zhou, J., and Yang, Y. (2022). “In-N-Out Generative Learning for Dense Unsupervised Video Segmentation”. In: *Proc. ACMM*.

Papazoglou, A. and Ferrari, V. (2013). “Fast object segmentation in unconstrained video”. In: *Proc. ICCV*.

Patriarche, J. and Erickson, B. (2004). “A review of the automated detection of change in serial imaging studies of the brain”. In: *Journal of digital imaging*.

Perazzi, F., Pont-Tuset, J., McWilliams, B., Van Gool, L., Gross, M., and Sorkine-Hornung, A. (2016). “A benchmark dataset and evaluation methodology for video object segmentation”. In: *Proc. CVPR*.

Petersen, F., Borgelt, C., Kuehne, H., and Deussen, O. (2021). “Differentiable sorting networks for scalable sorting and ranking supervision”. In: *Proc. ICML*.

Petersen, F., Borgelt, C., Kuehne, H., and Deussen, O. (2022). “Monotonic differentiable sorting networks”. In: *Proc. ICLR*.

Pickup, L. C. and Zisserman, A. (2009). “Automatic Retrieval of Visual Continuity Errors in Movies”. In: *Proc. CIVR*.

Ponimatkin, G., Samet, N., Xiao, Y., Du, Y., Marlet, R., and Lepetit, V. (2023). “A Simple and Powerful Global Optimization for Unsupervised Video Object Segmentation”. In: *Proc. WACV*.

Pont-Tuset, J., Perazzi, F., Caelles, S., Arbeláez, P., Sorkine-Hornung, A., and Gool, L. V. (2017). "The 2017 davis challenge on video object segmentation". In: *arXiv preprint arXiv:1704.00675*.

Qin, S., Bissacco, A., Raptis, M., Fujii, Y., and Xiao, Y. (2019). "Towards unconstrained end-to-end text spotting". In: *Proc. ICCV*.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. (2021). "Learning transferable visual models from natural language supervision". In: *arXiv preprint arXiv:2103.00020*.

Ramachandran, V. (1985). "Guest editorial: The neurobiology of perception". In: *Perception*.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. (2021). "Zero-Shot Text-to-Image Generation". In: *arXiv preprint arXiv:2102.12092*.

Ranftl, R. and Koltun, V. (2018). "Deep fundamental matrix estimation". In: *Proc. ECCV*.

Ranjan, A., Jampani, V., Balles, L., Sun, D., Kim, K., Wulff, J., and Black, M. J. (2019). "Competitive Collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation". In: *Proc. CVPR*.

Ravi, N., Gabeur, V., Hu, Y.-T., Hu, R., Ryali, C., Ma, T., Khedr, H., Rädle, R., Rolland, C., Gustafson, L., Mintun, E., Pan, J., Alwala, K. V., Carion, N., Wu, C.-Y., Girshick, R., Dollár, P., and Feichtenhofer, C. (2024). "Sam 2: Segment anything in images and videos". In: *arXiv preprint arXiv:2408.00714*.

Rocco, I., Arandjelovic, R., and Sivic, J. (2018). "End-to-end weakly-supervised semantic alignment". In: *Proc. CVPR*.

Ronneberger, O., Fischer, P., and Brox, T. (2015). "U-net: Convolutional networks for biomedical image segmentation". In: *Proc. MICCAI*.

Sachdeva, R. and Zisserman, A. (2023). "The Change You Want to See". In: *Proc. WACV*.

Safadoust, S. and Güney, F. (2023). "Multi-Object Discovery by Low-Dimensional Object Motion". In: *Proc. ICCV*.

Saha, S., Bovolo, F., and Bruzzone, L. (2019). "Unsupervised deep change vector analysis for multiple-change detection in VHR images". In: *IEEE Transactions on Geoscience and Remote Sensing*.

Sakurada, K. and Okatani, T. (2015). "Change Detection from a Street Image Pair using CNN Features and Superpixel Segmentation". In: *Proc. BMVC*.

Salomon, G., Laroca, R., and Menotti, D. (2020). “Deep learning for image-based automatic dial meter reading: Dataset and baselines”. In: *IJCNN*.

Sand, P. and Teller, S. (2008). “Particle video: Long-range motion estimation using point trajectories”. In: *IJCV*.

Scahill, R. I., Frost, C., Jenkins, R., Whitwell, J. L., Rossor, M. N., and Fox, N. C. (2003). “A longitudinal study of brain volume changes in normal aging using serial registered magnetic resonance imaging”. In: *Archives of neurology*.

Sevilla-Lara, L., Zha, S., Yan, Z., Goswami, V., Feiszli, M., and Torresani, L. (2021). “Only time can tell: Discovering temporal data for temporal modeling”. In: *Proc. WACV*.

Shi, B., Wang, X., Lyu, P., Yao, C., and Bai, X. (2016). “Robust scene text recognition with automatic rectification”. In: *Proc. CVPR*.

Shi, S., Gong, B., Chen, X., Zheng, D., Tan, S., Yang, Z., Li, Y., He, J., Zheng, K., Chen, J., Yang, M., and Zheng, Y. (2024). “MotionStone: Decoupled Motion Intensity Modulation with Diffusion Transformer for Image-to-Video Generation”. In: *arXiv preprint arXiv:2412.05848*.

Shrivastava, G., Lim, S.-N., and Shrivastava, A. (2024). “Video decomposition prior: Editing videos layer by layer”. In: *Proc. ICLR*.

Shvetsova, N., Petersen, F., Kukleva, A., Schiele, B., and Kuehne, H. (2023). “Learning by Sorting: Self-supervised Learning with Group Ordering Constraints”. In: *Proc. ICCV*.

Sidenbladh, H., Black, M. J., and Fleet, D. J. (2000). “Stochastic tracking of 3D human figures using 2D image motion”. In: *Proc. ECCV*.

Simonyan, K. and Zisserman, A. (2014). “Two-Stream Convolutional Networks for Action Recognition in Videos”. In: *NeurIPS*.

Sivic, J., Schaffalitzky, F., and Zisserman, A. (2004). “Object Level Grouping for Video Shots”. In: *Proc. ECCV*. Springer-Verlag.

Song, H., Wang, W., Zhao, S., Shen, J., and Lam, K.-M. (2018). “Pyramid dilated deeper convlstm for video salient object detection”. In: *Proc. ECCV*.

Soni, S., Dudhane, A., Debary, H., Fiaz, M., Munir, M. A., Danish, M. S., Fraccaro, P., Watson, C. D., Klein, L. J., Khan, F. S., and Khan, S. (2024). “EarthDial: Turning Multi-sensory Earth Observations to Interactive Dialogues”. In: *arXiv preprint arXiv:2412.15190*.

Srinivasan, P. P., Tucker, R., Barron, J. T., Ramamoorthi, R., Ng, R., and Snavely, N. (2019). “Pushing the Boundaries of View Extrapolation with Multiplane Images”. In: *Proc. CVPR*.

Stent, S., Gherardi, R., Stenger, B., and Cipolla, R. (2015). "Detecting Change for Multi-View, Long-Term Surface Inspection". In: *Proc. BMVC*.

Sun, D., Sudderth, E. B., and Black, M. J. (2012). "Layered segmentation and optical flow estimation over time". In: *Proc. CVPR*. IEEE.

Sun, D., Wulff, J., Sudderth, E., Pfister, H., and Black, M. J. (2013). "A fully-connected layered model of foreground and background flow". In: *Proc. CVPR*.

Sun, D., Yang, X., Liu, M.-Y., and Kautz, J. (2018). "PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume". In: *Proc. CVPR*.

Sun, Q., Cui, Y., Zhang, X., Zhang, F., Yu, Q., Wang, Y., Rao, Y., Liu, J., Huang, T., and Wang, X. (2024). "Generative multimodal models are in-context learners". In: *Proc. CVPR*.

Sun, Y., Chen, J., Zhang, S., Zhang, X., Chen, Q., Zhang, G., Ding, E., Wang, J., and Li, Z. (2024). "VRP-SAM: SAM with Visual Reference Prompt". In: *Proc. CVPR*.

Svennerholm, L., Boström, K., and Jungbjer, B. (1997). "Changes in weight and compositions of major membrane components of human brain during the span of adult human life of Swedes". In: *Acta neuropathologica*.

Szeliski, R. (2004). *Image Alignment and Stitching: A Tutorial*. Tech. rep. URL: https://www.microsoft.com/en-us/research/publication/image-alignment-and-stitching-a-tutorial/.

Tang, L., Xiao, H., and Li, B. (2023). "Can SAM Segment Anything? When SAM Meets Camouflaged Object Detection". In: *arXiv preprint arXiv:2304.04709*.

Teed, Z. and Deng, J. (2020). "RAFT: Recurrent All-Pairs Field Transforms for Optical Flow". In: *Proc. ECCV*.

Tokmakov, P., Alahari, K., and Schmid, C. (2017). "Learning motion patterns in videos". In: *Proc. CVPR*.

Tokmakov, P., Schmid, C., and Alahari, K. (2019). "Learning to segment moving objects". In: *IJCV*.

Ulyanov, D., Vedaldi, A., and Lempitsky, V. (2016). "Instance normalization: The missing ingredient for fast stylization". In: *arXiv preprint arXiv:1607.08022*.

Van Etten, A., Hogan, D., Manso, J. M., Shermeyer, J., Weir, N., and Lewis, R. (2021). "The multi-temporal urban development spacenet dataset". In: *Proc. CVPR*.

Vara, V. S. (2021). *Reading analog clocks with neural networks*. https://github.com/VictorSuarezVara/Reading-analog-clocks-with-neural-networks.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). "Attention is All you Need". In: *NeurIPS*.

Ventura, C., Bellver, M., Girbau, A., Salvador, A., Marques, F., and Giro-i-Nieto, X. (2019). "RVOS: End-to-End Recurrent Network for Video Object Segmentation". In: *Proc. CVPR*.

Vinyals, O., Fortunato, M., and Jaitly, N. (2015). "Pointer networks". In: *NeurIPS*.

Voigtlaender, P., Chai, Y., Schroff, F., Adam, H., Leibe, B., and Chen, L.-C. (2019). "Feelvos: Fast end-to-end embedding learning for video object segmentation". In: *Proc. CVPR*.

Voigtlaender, P. and Leibe, B. (2017). "Online adaptation of convolutional neural networks for video object segmentation". In: *Proc. BMVC*.

Vondrick, C., Shrivastava, A., Fathi, A., Guadarrama, S., and Murphy, K. (2018). "Tracking emerges by colorizing videos". In: *Proc. ECCV*.

Wang, J., Jiao, J., and Liu, Y. (2020). "Self-Supervised Video Representation Learning by Pace Prediction". In: *Proc. ECCV*.

Wang, J. Y. and Adelson, E. H. (1994). "Representing moving images with layers". In: *IEEE transactions on image processing*.

Wang, W., Lu, X., Shen, J., Crandall, D. J., and Shao, L. (2019). "Zero-shot video object segmentation via attentive graph neural networks". In: *Proc. ICCV*.

Wang, W., Shen, J., Yang, R., and Porikli, F. (2018). "Saliency-Aware Video Object Segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Wang, X., Girshick, R., Gupta, A., and He, K. (2018). "Non-local Neural Networks." In: *Proc. CVPR*.

Wang, X., Jabri, A., and Efros, A. A. (2019). "Learning correspondence from the cycle-consistency of time". In: *Proc. CVPR*.

Wang, X., Misra, I., Zeng, Z., Girdhar, R., and Darrell, T. (2023). "VideoCutLER: Surprisingly Simple Unsupervised Video Instance Segmentation". In: *arXiv preprint arXiv:2308.14710*.

Wei, D., Lim, J. J., Zisserman, A., and Freeman, W. T. (2018). "Learning and using the arrow of time". In: *Proc. CVPR*.

Wei, S.-E., Ramakrishna, V., Kanade, T., and Sheikh, Y. (2016). "Convolutional pose machines". In: *Proc. CVPR*.

Wertheimer, M. (1923). "Untersuchungen zur Lehre von der Gestalt. II". In: *Psychologische forschung*.

Wu, J., Ji, W., Liu, Y., Fu, H., Xu, M., Xu, Y., and Jin, Y. (2023). "Medical SAM Adapter: Adapting Segment Anything Model for Medical Image Segmentation". In: *arXiv preprint arXiv:2304.12620*.

Wulff, J. and Black, M. J. (2014). "Modeling Blurred Video with Layers". In: *Proc. ICCV*.

Wulff, J. and Black, M. J. (2015). "Efficient Sparse-to-Dense Optical Flow Estimation using a Learned Basis and Layers". In: *Proc. ECCV*.

Xie, C., Xiang, Y., Harchaoui, Z., and Fox, D. (2019). "Object discovery in videos as foreground motion clustering". In: *Proc. CVPR*.

Xie, J., Xie, W., and Zisserman, A. (2022). "Segmenting Moving Objects via an Object-Centric Layered Representation". In: *NeurIPS*.

Xie, J., Xie, W., and Zisserman, A. (2024). "Appearance-based Refinement for Object-Centric Motion Segmentation". In: *Proc. ECCV*.

Xiong, Y., Varadarajan, B., Wu, L., Xiang, X., Xiao, F., Zhu, C., Dai, X., Wang, D., Sun, F., Iandola, F., Krishnamoorthi, R., and Chandra, V. (2023). "EfficientSAM: Leveraged Masked Image Pretraining for Efficient Segment Anything". In: *arXiv:2312.00863*.

Xu, N., Yang, L., Fan, Y., Yue, D., Liang, Y., Yang, J., and Huang, T. (2018). "Youtube-vos: A large-scale video object segmentation benchmark". In: *Proc. ECCV*.

Xue, T., Rubinstein, M., Liu, C., and Freeman, W. T. (2015). "A Computational Approach for Obstruction-Free Photography". In: *SIGGRAPH*.

Yang, C., Lamdouar, H., Lu, E., Zisserman, A., and Xie, W. (2021). "Self-supervised Video Object Segmentation by Motion Grouping". In: *Proc. ICCV*.

Yang, C., Xie, W., and Zisserman, A. (2022). "It's About Time: Analog Clock Reading in the Wild". In: *Proc. CVPR*.

Yang, S., Zhang, L., Qi, J., Lu, H., Wang, S., and Zhang, X. (2021). "Learning Motion-Appearance Co-Attention for Zero-Shot Video Object Segmentation". In: *Proc. ICCV*.

Yang, Y., Lai, B., and Soatto, S. (2021). "DyStaB: Unsupervised Object Segmentation via Dynamic-Static Bootstrapping". In: *Proc. CVPR*.

Yang, Y., Loquercio, A., Scaramuzza, D., and Soatto, S. (2019). "Unsupervised Moving Object Detection via Contextual Information Separation". In: *Proc. CVPR*.

Yang, Z., Wang, Q., Bertinetto, L., Bai, S., Hu, W., and Torr, P. H. (2019). "Anchor Diffusion for Unsupervised Video Object Segmentation". In: *Proc. ICCV*.

Yang, Z. and Yang, Y. (2022). “Decoupling Features in Hierarchical Propagation for Video Object Segmentation”. In: *NeurIPS*.

Ye, V., Li, Z., Tucker, R., Kanazawa, A., and Snavely, N. (2022). “Deformable sprites for unsupervised video decomposition”. In: *Proc. CVPR*.

Yeung, J., Luo, A. F., Sarch, G., Henderson, M. M., Ramanan, D., and Tarr, M. J. (2024). “Reanimating Images using Neural Representations of Dynamic Stimuli”. In: *arXiv preprint arXiv:2406.02659*.

Zarrabi, N., Avidan, S., and Moses, Y. (2018). “Crowdcam: Dynamic region segmentation”. In: *arXiv preprint arXiv:1811.11455*.

Zhang, C., Han, D., Qiao, Y., Kim, J. U., Bae, S.-H., Lee, S., and Hong, C. S. (2023). “Faster Segment Anything: Towards Lightweight SAM for Mobile Applications”. In: *arXiv preprint arXiv:2306.14289*.

Zhang, K., Zhao, Z., Liu, D., Liu, Q., and Liu, B. (2021). “Deep Transport Network for Unsupervised Video Object Segmentation”. In: *Proc. ICCV*.

Zhang, P., Yang, Z., Yu, H., Tu, W., Gao, C., and Wang, Y. (2024). “RUSNet: Robust fish segmentation in underwater videos based on adaptive selection of optical flow”. In: *Frontiers in Marine Science*.

Zhang, P., Yu, H., Li, H., Zhang, X., Wei, S., Tu, W., Yang, Z., Wu, J., and Lin, Y. (2023). “Msgnet: multi-source guidance network for fish segmentation in underwater videos”. In: *Frontiers in Marine Science*.

Zhang, X., Gu, C., and Zhu, S. (2023). “SAM-helps-Shadow:When Segment Anything Model meet shadow removal”. In: *arXiv preprint arXiv:2306.06113*.

Zhang, Z., Zhang, S., Wei, Z., Dai, Z., and Zhu, S. (2024). “UVOSAM: A Mask-free Paradigm for Unsupervised Video Object Segmentation via Segment Anything Model”. In: *arXiv preprint arXiv:2305.12659*.

Zhao, X., Ding, W., An, Y., Du, Y., Yu, T., Li, M., Tang, M., and Wang, J. (2023). “Fast Segment Anything”. In: *arXiv preprint arXiv:2306.12156*.

Zheng, Z., Zhong, Y., Zhang, L., and Ermon, S. (2024). “Segment Any Change”. In: *NeurIPS*.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. (2016). “Learning deep features for discriminative localization”. In: *Proc. CVPR*.

Zhou, T., Wang, S., Zhou, Y., Yao, Y., Li, J., and Shao, L. (2020). “Motion-attentive transition for zero-shot video object segmentation”. In: *AAAI*.

Zhou, T., Brown, M., Snavely, N., and Lowe, D. G. (2017). “Unsupervised learning of depth and ego-motion from video”. In: *Proc. CVPR*.

Zhou, T., Tucker, R., Flynn, J., Fyffe, G., and Snavely, N. (2018). “Stereo Magnification: Learning View Synthesis using Multiplane Images”. In: *SIGGRAPH*.

Zhukov, D., Alayrac, J.-B., Laptev, I., and Sivic, J. (2020). “Learning actionness via long-range temporal order verification”. In: *Proc. ECCV*.

Zitnick, C. L., Kang, S. B., Uyttendaele, M., Winder, S., and Szeliski, R. (2004). “High-quality video view interpolation using a layered representation”. In: *ACM transactions on graphics (TOG)*.

Zou, X., Yang, J., Zhang, H., Li, F., Li, L., Gao, J., and Lee, Y. J. (2023). “Segment everything everywhere all at once”. In: *NeurIPS*.

Appendix A

Statement of authorship

A statement of authorship is provided for each multi-authored paper included in this thesis. The statements describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication, there exists a complete statement that is filled out and signed by the candidate and supervisor.

Statement of Authorship for the paper "Betrayed by Motion: Camouflaged object discovery via motion segmentation" in Chapter 2.

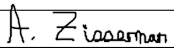
Paper title	Betrayed by Motion: Camouflaged object discovery via motion segmentation
Authors	Hala Lamdouar, Charig Yang, Weidi Xie, Andrew Zisserman
Publication status	Published
Publication details	Asian Conference on Computer Vision (ACCV), 2020.

Student Confirmation

Student name	Charig Yang	
Contribution to the paper	Second-author contribution: • dataset collection and labelling • writing and presentation of the paper	
Signature and Date		Jan 9th 2025

Supervisor Confirmation

By signing the Statement of Authorship, the supervisor is certifying that the candidate made a substantial contribution to the publication, and that the description above is accurate.

Supervisor name	Prof. Andrew Zisserman	
Supervisor comments		
Signature and Date		Jan 9th 2025

Statement of Authorship for the paper "Self-supervised video object segmentation by motion grouping" in Chapter 3.

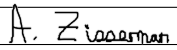
Paper title	Self-supervised video object segmentation by motion grouping
Authors	Charig Yang, Hala Lamdouar, Erika Lu, Andrew Zisserman, Weidi Xie
Publication status	Published
Publication details	International Conference on Computer Vision (ICCV), 2021.

Student Confirmation

Student name	Charig Yang	
Contribution to the paper	First-author contribution: • conception of research ideas • design and implementation of models • writing and presentation of the paper	
Signature and Date		Jan 9th 2025

Supervisor Confirmation

By signing the Statement of Authorship, the supervisor is certifying that the candidate made a substantial contribution to the publication, and that the description above is accurate.

Supervisor name	Prof. Andrew Zisserman	
Supervisor comments		
Signature and Date		Jan 9th 2025

Statement of Authorship for the paper "Moving Object Segmentation: All you need is SAM (and flow)" in Chapter 4.

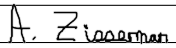
Paper title	Moving Object Segmentation: All you need is SAM (and flow)
Authors	Junyu Xie, Charig Yang, Weidi Xie, Andrew Zisserman
Publication status	Published
Publication details	Asian Conference on Computer Vision (ACCV), 2024.

Student Confirmation

Student name	Charig Yang	
Contribution to the paper	Second-author contribution: • partial conception of research ideas • partial design and implementation of models • writing and presentation of the paper	
Signature and Date		Jan 9th 2025

Supervisor Confirmation

By signing the Statement of Authorship, the supervisor is certifying that the candidate made a substantial contribution to the publication, and that the description above is accurate.

Supervisor name	Prof. Andrew Zisserman	
Supervisor comments		
Signature and Date		Jan 9th 2025

Statement of Authorship for the paper "It's About Time: Analog clock reading in the wild" in Chapter 5.

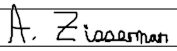
Paper title	It's About Time: Analog clock reading in the wild
Authors	Charig Yang, Weidi Xie, Andrew Zisserman
Publication status	Published
Publication details	Conference on Computer Vision and Pattern Recognition (CVPR), 2022.

Student Confirmation

Student name	Charig Yang	
Contribution to the paper	First-author contribution: • conception of research ideas • design and implementation of models • writing and presentation of the paper	
Signature and Date		Jan 9th 2025

Supervisor Confirmation

By signing the Statement of Authorship, the supervisor is certifying that the candidate made a substantial contribution to the publication, and that the description above is accurate.

Supervisor name	Prof. Andrew Zisserman	
Supervisor comments		
Signature and Date		Jan 9th 2025

Statement of Authorship for the paper "Made to Order: Discovering monotonic temporal changes via self-supervised video ordering" in Chapter 6.

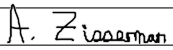
Paper title	Made to Order: Discovering monotonic temporal changes via self-supervised video ordering
Authors	Charig Yang, Weidi Xie, Andrew Zisserman
Publication status	Published
Publication details	European Conference on Computer Vision (ECCV), 2024.

Student Confirmation

Student name	Charig Yang	
Contribution to the paper	First-author contribution: • conception of research ideas • design and implementation of models • writing and presentation of the paper	
Signature and Date		Jan 9th 2025

Supervisor Confirmation

By signing the Statement of Authorship, the supervisor is certifying that the candidate made a substantial contribution to the publication, and that the description above is accurate.

Supervisor name	Prof. Andrew Zisserman	
Supervisor comments		
Signature and Date		Jan 9th 2025