Video Models of People and Pixels

Jathushan Rajasegaran

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2025-65
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2025/EECS-2025-65.html

May 15, 2025

Acknowledgement

I thank my advisor Jitendra Malik for guiding me through this wonderful PhD journey. I thank my committee members Angjoo Kanazawa, Alyosha Efros and Bruno Olshausen for giving valuable feedback over the years and helping me with research. Finally I thank my parents for their unconditional love and support.

Video Models of People and Pixels

By

Jathushan Rajasegaran

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering - Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Arthur J. Chick Professor Jitendra Malik, Chair
Assistant Professor Angjoo Kanazawa
Howard Friesen Professor Alexei (Alyosha) Efros
Professor Bruno Olshausen

Spring 2025

Video Models of People and Pixels

Abstract

Video Models of People and Pixels

by

Jathushan Rajasegaran

Doctor of Philosophy in Engineering - Electrical Engineering and Computer Sciences

University of California, Berkeley

Arthur J. Chick Professor Jitendra Malik, Chair

From the moment we are born, we continuously witness the "video" of our own lives—hundreds of thousands of hours of rich, unfolding scenes. These visual experiences, streaming in seamlessly over time, form the foundation of how we understand the world: by tracking motion, recognizing people, and anticipating what comes next. In many ways, our perception begins with tracking—following a pixel, a person, or a motion—enabling higher-order understanding such as object permanence, social interaction, and physical causality. This thesis explores how to build visual models that can track, recognize, and predict.

First I will discuss about tracking people in monocular videos with PHALP (Predicting Human Appearance, Location, and Pose). By aggregating 3D representations into tracklets, temporal models predict future states, enabling persistent tracking. Next, I will discuss human action recognition from a Lagrangian perspective using these tracklets. LART (Lagrangian Action Recognition with Tracking), a transformer-based model, demonstrates the benefits of explicit 3D pose (SMPL) and location for predicting actions. LART fuses 3D pose dynamics with contextualized appearance features along tracklets, significantly improving performance on the AVA dataset, especially for interactive and complex actions. Finally, I will discuss about large-scale self-supervised learning through autoregressive video prediction with Toto, a family of causal transformers. Trained on next-token prediction using over a trillion visual tokens from diverse image and video datasets, Toto learns powerful, general-purpose visual representations with minimal inductive biases. An empirical study of architectural and tokenization choices shows these representations achieve competitive performance on downstream tasks including classification, tracking, object permanence, and robotics. We also analyze the power-law scaling of these video models.

To Appa and Amma
for your unconditional love and support

Contents

Acknowledgments

This dissertation wouldn't have been possible without the incredible people who've been part of this journey. What follows is just a small attempt to thank them for the constant support, encouragement, and inspiration they've given me along the way.

My advisor, Jitendra Malik. I still remember the first call we had during admissions—it was a turning point in my life. It is really hard to put into words how grateful I am to you for giving me these amazing five years at Berkeley. You taught me so many things: how to run proper ablations, how to write a good introduction, how to give a talk, how to think about long-term research, and countless other pieces of advice I will continue to realize over time. I will forever cherish our long weekend meetings; they are some of the best memories of my life. Thanks for all the book recommendations and gelatos. You are the reason I found amazing friends, great collaborators, beautiful memories, and all the research outcomes—thank you, Jitendra, for everything.

The welcoming culture at Berkeley enabled so many great interactions and collaborations. I was so fortunate to work with Angjoo Kanazawa, and forever grateful to her for finding my application. It was a dark time during COVID, but working with Angjoo somehow balanced it, her energy and positivity made me productive in research. Alyosha Efros, thanks for all the late-night discussions and buying food for the late-night crew, and of course, taking us to all the beautiful places in Berkeley and SF. I would cherish all the hikes and the fun conversations we had during the hikes and gelatos. Thanks, Alyosha, for making this journey a beautiful one. Bruno Olshausen, I met him first during my quals, from that point onward he is so kind to me, always pushing me and asking challenging questions. Thanks Bruno for making this dissertation a memorable one for me. I was also fortunate to work with other faculty members in Berkeley - Trevor Darrell and Sergey Levine. It was such a great experience working with them, and thanks for making BAIR an amazing place for collaboration.

Georgios Pavlakos is the first person I go to whenever I have a problem, whether it is about SMPL mesh or about life. He has been a great friend, collaborator, and mentor. I am grateful for his support during my early years. Without him, I cannot imagine how my first years would have been—or how I would have rotated a mesh without him pointing me to the right codebase. My last years at Meta were made beautiful thanks to Christoph Feichtenhofer. Thanks for letting me explore new ideas and all of the actionable feedback.

My undergraduate advisor Ranga Rodrigo, thanks for finding me and pushing me to do research during my undergrad days. He guide me through tough times and was always there whenever I needed him, Thank you Ranga Sir for everything. Suranga Seneviratne gave me an internship at dat61, and I am forever grateful for this amazing opportunity. Chamira Edusooiya and Tharaka Samarasinghe were so supportive of me during my time at Moratuwa and after, thank you so much. Kirthevasan Kandasamy was the first person I started doing research with, a fun fact, it was about style transfer. Thanks, Kandasamy, for finding me and pushing me to do research during my undergrad days. I wrote my first paper with Sameera Ramasinghe. Thanks for allowing me to explore and try new things during this time. After my undergraduate, I was bit lost, Salman Khan, gave me a hand and gave me a wonderful opportunity work with him and give me the freedom to do research and explore new ideas. Thanks Salman for being there for me, when I needed the most.

Beyond the support from my mentors, I've been lucky to work with some truly amazing collaborators, who have become friends forever. This thesis wouldn't exist without their hard work, energy, and constant push to keep going. I'm especially grateful to them for putting up with my flaws and always pushing me to do better. Shubham is a great friend and mentor—he taught me how to write good code. We used to stay up late at night and debug models, and play cricket. It was a pleasure to have crossed paths with Shub. Sasha is an amazing friend, a great co-op player, and sometimes even an in-house therapist. Thanks, Sasha, for helping me beat Elden Ring. Working with Ilija was a great experience, thanks for letting me play with robots and work on something out of my comfort zone, is a great learning curve. Thanks, Karthik, for taking me to nice places in Berkeley during COVID times and teaching me about transformers. Vongani is a great friend who always listens to me whenever I have problems; her calm nature always amazes me. Thanks, Von, for being a great friend and supporting me through hard times. Thanks, Himanshu, for coming with me on long walks and for the fruitful discussion on research. I only started working with Jane in my last semester, but she has been an amazing collaborator and a great friend. Her calm nature during the deadline time kept us calm. Boyi, working with her was such a fun experience. My desk mate, Lea—thank you for being there during tough times and for teaching me tennis. Yossi, been a great friend, had a great time working with him, teaching CS182, and going on hikes. Our last row in Vision Bay would not be complete without Antonio, we had so much fun telling jokes and talking about research. Another member of our last row vision bay is Neerja; her skills in organization and enthusiasm are unmatched. Amil, thanks for the thoughtful discussions on loss landscapes. Evonne she has been a great friend from day one! Thanks, Evonne, for laughing at my jokes. Vickie was there for me during the hard time. I was able to pass these days only because of the support of my friends. Ruilong is such an amazing friend, he would always help me whenever I have questions on how to install CUDA, always smiling, and thanks for driving me on long walks in Napa. Hang, thanks for all the interesting discussions on AI over our long walks in SF. Thanks Suzie and Medhini for managing to sit next to me everyday and enduring my bad jokes. Amir is another core member of our Vision Bay last row. Thanks for all the fun evenings at the gym and self-supervised learning discussions. It was a great experience working with Shiry, had amazing discussions on both real and artificial neurons. I know this is a long list, but these people made my life at Berkeley a beautiful memory. I am thankful to all of them forever.

To my undergrad students from Berkeley Rahul, thanks for choosing to work with me, we explored so many ideas together. It was such beautiful experience to see you grow from a student to researcher, I am glad I played a small part in your research carrier. Tanish, thanks for working with me last year, it was a great experience to try new ideas and explore with you.

To my meta mates, Andrea, Bernie, Danial, and Vincent, it was an amazing time working with them, sharing ideas, and having whiteboard sessions. Thank you, Xinlei, for exploring new directions and trying new ideas with me.

To my best friend Vinoj—it's been about two years without you. We wrote our first paper together, when to the first conference together, it is hard to imagine research without you. I would have never imagined writing this part in my thesis. You were a mentor, a friend, and a brother to me. Every CVPR will remind you. life without you is going to be hard, but I will carry the memories of your forever. Miss you man!

Thanks to the BAIR Admin team - Angie, Roxana, and Ami- for making BWW life pleasant and shielding me from bureaucracy and logistics, often without me even knowing it. Thanks to Wasim Younis, from BIO, who helped me so many times and always made sure that my applications are approved on time, thank you so much Wasim.

I am thankful to my housemates Adwait, Naman, and Aayan for managing to live with me for the last 5 years. Thanks for teaching me how to cook, thanks for the really long walks, and late-night gelatos. Thanks, Aayan and Arpita, for cooking me amazing food after deadlines and watching Marvel movies.

Hirunima, Athif and Sanduru, thanks for calling regularly and checking on me, listening to my problems and traveling with me. My Friends from Moratuwa: Priyanthan, Thuvakaran, Mathushan, Nilakshan, Thivakaran, Keerthanan, Sivaneeban, Nirukan, Sudeera, Danial, Ravindu, Shehan, Kasthuri, and Hasitha, thank you for being there for me, and it was a great experience to study and build with all of you.

This journey started well before my time at Berkeley, and I owe a lot to the many people who've shaped me along the way. Friends, teachers, and professors who have sparked my curiosity and love for learning early on, and I'm deeply grateful for the role they've played in who I am today.

My Tamil teacher, Mr. Thangavel, not only taught me the language but also lessons in life, kindness, and ethical living. Mr Murukavel, the first person who sparked my scientific curiosity, to look for the stars! We made telescopes, we built a small lab in my house to do many experiments. Mr. Ladchumanan, my Math teacher—fun fact: he can write with both hands at the same time—challenged me to push my limits and helped me become a faster thinker. Mr. Sothilingam, my Physics teacher, always encouraged logical thinking and curiosity. And Mr. Mukunthan, my English teacher, truly cared about me—he still calls to check in and see how I'm doing, thank you.

To my school friends[1] — Gowsi, Vibishanth, Kuruparan, Guruparan, Ramraj, Athavaloshan, Athavan, Santhoshan, Parthipan, Sajinthan, Sarma, Tharsan, Suthagar, and Vennilavan—thank you for being there for me at different points in my life. You've helped me tackle challenges, both on paper and in life. Grateful for all of you!

To my parents, there aren't enough words big enough to capture how grateful I am. Your love, sacrifices, and unshakable belief in me have been the foundation of everything. You taught me what resilience, humility, and hard work truly mean. Vadivambikai Rajasegaran—Amma, I love you so much! You have cared for me from the day I was born, you left your job to take care of me, you moved with me whenever I moved to a new school, leaving behind your home, family, and friends. Rajasegaran Balakrishar– Appa, my first teacher, you sacrificed so much for me, for years you took me on your bicycle all over Jaffna, hiding all the pain, spending all the money we had for my education. Love you, Amma, Appa!

What am I, if not sculpted by my teachers? Everyone mentioned above was at some point a teacher to me. I credit everything I have done and will do to all my teachers.

[1] Sorry if I forgot to thank someone. If you know me, you know I am a bit of a forgetful person

Chapter 1

Introduction

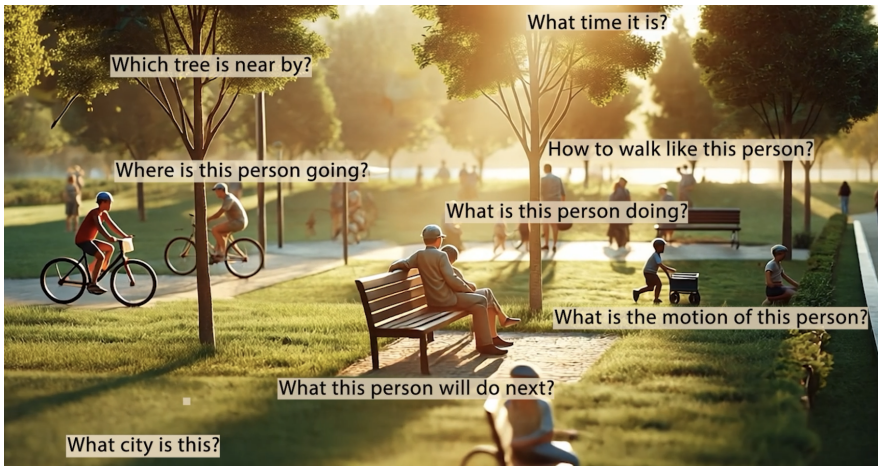


Figure 1.1: **An Evening Walk:** We go on evening walk, near Berkeley, we see a nice park and people doing various activities. Just by looking at this video, we can answer many questions about the video. *which tree is near to us?, where is the bicyclist going?, what the person on the bench doing? etc.*

.

It is springtime in Berkeley. The days are warm and golden, and the sun sets slowly around 8 p.m. I often go for a walk toward Albany. On my way, I see people heading home after work, long queues forming outside cheese board pizza, and on Solano street restaurants are buzzing with energy. The sidewalks are lively—strollers, cyclists, couples, dogs, and laughter. Eventually, I pass by a small park—something like the one shown in Fig. 1.1. I sit down and watch the sunset. I see people are doing so many activities. By watching this seen for a while (*'a video'*), I can answer all the questions about this video. From *which tree is near to us?, where is the bicyclist going?, what the person on the bench doing? what time it could be? how to imitate a walking style of a person?*

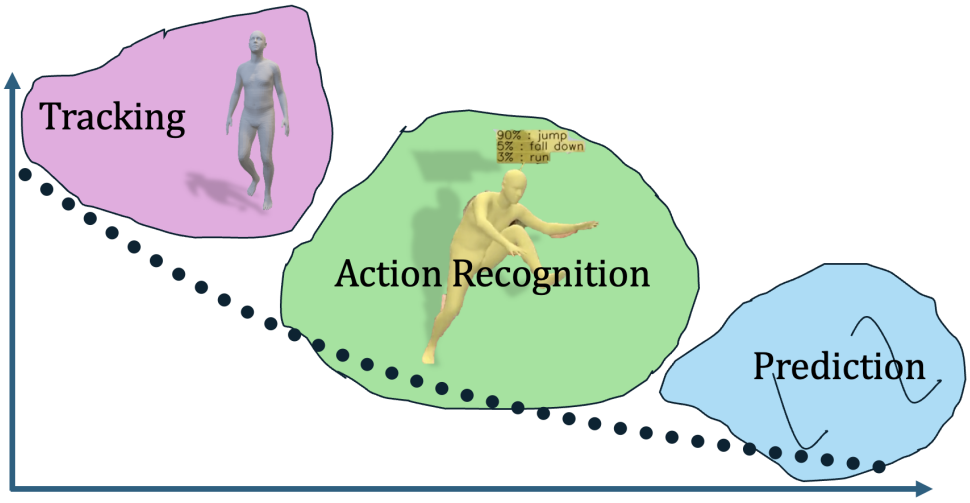


Figure 1.2: **The structure of this thesis:** In this thesis, we will look into these three problems, tracking, action recognition and prediction. We will look at them and see how each problem could connected to other and how they need more compute (x-axis) as we remove inductive bias (y-axis).

what the kid near the tree would do next? etc. We could say all of these questions are part of the universal problem 'learning from videos' or 'video understanding'.

In this thesis, we are going to define video understanding as the **gain in information after watching a video.** This could be humans watching a video and their ability to answer many questions about the video, or at the model inference the change of model activations or changed weights at train time. We will treat all of these as an understanding of the video.

All the questions we see in the Fig 1.1, has been studied under various areas, for example: *where is the bicyclist going?* is a tracking problem, *what the person on the bench doing?* is a action recognition problem, *how to imitate a walking style of a person?* could be robotics problem, *what the kid near the tree would do next?* is a prediction problem. In this thesis, we will cover 3 such problems, **tracking, action recognition and prediction**. Fig 1.2 shows these three problems with increasing scale of compute (x-axis). We can thing this in terms of going from 3D models (tracking) to fusing 3D+2D (action recognition) to fully 2D based approach (prediction). We can look at them as human centric models (tracking) to more general model. Also, very well defined problems (tracking is a very well defined problem) to no so well defined evaluation of the prediction models (apart from the prediction task, rest are not so well defined.)

Tracking Humans in 3D [111, 110]: The first part of this thesis introduces PHALP (Predicting Human Appearance, Location, and Pose), a tracking system designed for monocular video. Unlike conventional 2D tracking approaches, PHALP reasons directly in 3D space by lifting single-frame detections into 3D representations using SMPL-based models. These representations capture not only where a person is but also how they look and move over time. By aggregating this information into tracklets, PHALP builds dynamic models for each identity and predicts their future states.

This predictive capability—across appearance, pose, and location—enables PHALP to maintain identity across frames, even through occlusions and shot transitions. PHALP incorporates learned appearance embeddings based on 3D texture map, linearized motion prediction, and transformer-based pose forecasting. It achieves state-of-the-art results on several benchmarks and demonstrates how a good 3D representation can simplifies identity persistence in complex, real-world scenes.

Recognizing Actions over Trajectories [109]: Once we can reliably track individuals in 3D, we can ask richer questions: What is this person doing? Who are they interacting with? Can we predict their future actions?

To this end, the second part of this thesis introduces LART (Lagrangian Action Recognition with Tracking), a transformer-based model that treats each human trajectory as a temporally evolving entity. Instead of analyzing videos from a fixed viewpoint—as most image- or grid-based video models do—LART adopts a Lagrangian perspective, following each person through time and fusing their 3D pose and appearance across their trajectory. This person-centric representation allows LART to reason about actions as high-level dynamics rather than as local patterns.

The model is particularly powerful for recognizing interactive actions, such as “hugging", "dancing", or "kissing," which often involve subtle temporal dependencies and spatial context. On the AVA dataset, LART achieves significant improvements over prior methods, especially in difficult action classes that require understanding motion, pose, and human-object interaction. It also demonstrates that combining geometry (pose) and semantics (appearance) in a temporally structured way leads to more robust and interpretable models.

Learning from Massive Video Data [108]: The final part of this thesis explores how large-scale, self-supervised video models can be trained using simple objectives. We introduce Toto, a family of causal transformer models trained via next-token prediction on over a trillion visual tokens from videos and images. Unlike typical supervised video models, Toto makes minimal assumptions about inductive biases. It is trained in a fully autoregressive manner, learning to predict the next patch.

We study how design choices—such as tokenization granularity, frame sampling, and model architecture—affect the emergence of useful representations. Toto demonstrates strong performance on a wide range of downstream tasks, including video classification, tracking, object permanence, and robotics—often matching or exceeding more task-specific models. Perhaps more importantly, the simplicity of the learning objective reveals clear power-law scaling behavior (which closer but is still slower than language), suggesting a path forward for training ever larger and more general video models.

Summary: The three parts of this thesis offer a coherent approach to video understanding—starting from precise tracking, moving to structured recognition, and finally to large-scale predictive learning. The core insight is that temporality matters: effective video models must reason not only about what is visible but *how it changes, persists, and unfolds over time*. By grounding these models in 3D human-centric representations and scaling them with minimal supervision.

In the chapters that follow, we develop each of these ideas in detail, supported by empirical results, and open-source systems. I hope is that this work contributes to the broader goal of building better video models.

Chapter 2

Tracking People by Predicting 3D Appearance, Location and Pose

We present an approach for tracking people in monocular videos by predicting their future 3D representations. To achieve this, we first lift people to 3D from a single frame in a robust manner. This lifting includes information about the 3D pose of the person, their location in the 3D space, and the 3D appearance. As we track a person, we collect 3D observations over time in a tracklet representation. Given the 3D nature of our observations, we build temporal models for each one of the previous attributes. We use these models to predict the future state of the tracklet, including 3D appearance, 3D location, and 3D pose. For a future frame, we compute the similarity between the predicted state of a tracklet and the single frame observations in a probabilistic manner. Association is solved with simple Hungarian matching, and the matches are used to update the respective tracklets. We evaluate our approach on various benchmarks and report state-of-the-art results. Code and models are available at: `https://brjathu.github.io/PHALP`.

2.1 Introduction

When we watch a video, we can segment out individual people, cars, or other objects and track them over time. The corresponding task in computer vision has been studied for several decades now, with a fundamental choice being whether to do the tracking in 2D in the image plane, or of 3D objects in the world. The former seems simpler because it obviates the need for inferring 3D, but if we do take the step of back-projecting from the image to the world, other aspects such as dealing with occlusion become easier. In the 3D world the tracked object doesn't disappear, and even young infants are aware of its persistence behind the occluder. In our recent work [111], we presented experimental evidence that performance is better with 3D representations. In this chapter, we will take this as granted, and proceed to develop a system in the 3D setting of the problem. While our approach broadly applies to any object category where parameterized 3D models are available and can be inferred from images, we will limit ourselves in this chapter to studying people, the most important case in practice.

Figure 2.1: **Tracking people by predicting and matching in 3D:** The top row shows our tracking results at three different frames. The results are visualized by a colored head-mask for unique identities. The second and third rows show renderings of the 3D states of the two people in their associated tracklets. The bottom row shows the bottom-up detections in each image frame which, after being lifted to 3D, will be matched with the 3D predictions of each tracklet in the corresponding frame. Note how in the middle frame of second row, the 3D representation of the person persists even though they are occluded in the image. Readers are encouraged to watch the videos at the project website.

Once we have accepted the philosophy that we are tracking 3D objects in a 3D world, but from 2D images as raw data, it is natural to adopt the vocabulary from control theory and estimation theory going back to the 1960s. We are interested in the "state" of objects in 3D, but all we have access to are "observations" which are RGB pixels in 2D. In an online setting, we observe a person across multiple time frames, and keep recursively updating our estimate of the person's state — their appearance, location in the world, and pose (configuration of joint angles). Since we have a dynamic model (a "tracklet"), we can also predict states at future times. When the next image frame comes in, we detect the people in it, lift them to 3D, and in that setting solve the association problem between these bottom-up detections and the top-down predictions of the different tracklets for this

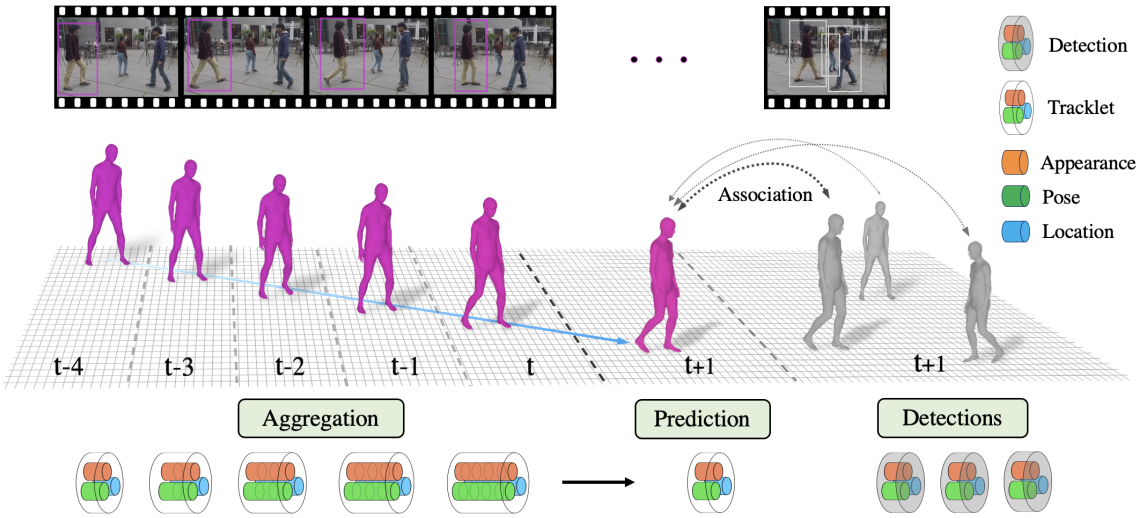


Figure 2.2: **PHALP: Predicting Human Appearance, Location and Pose for Tracking**: We perform tracking of humans in 3D from monocular video. For every input bounding box, we estimate a 3D representation based on the 3D appearance, 3D pose and 3D location of the person. During tracking, these are integrated to form corresponding tracklet-based representations. We perform tracking by predicting the future representation of each person and using it to solve for association given the detected bounding boxes of a future frame.

frame. Once the observations have been associated with the tracklets, the state of each person is re-estimated and the process continues. Fig. 2.1 shows this process on a real video. Note that during a period of occlusion of a tracklet, while no new observations are coming in, the state of the person keeps evolving following their dynamics. It is not the case that "Out of sight, out of mind"!

In an abstract form, the procedure sketched in the previous paragraph is basically the same as that followed in multiple computer vision papers from the 1980s and 1990s. The difference is that in 2022 we can actually make it work thanks to the advances brought about by deep learning and big data, that enable consistent and reliable lifting of people to 3D. For this initial lifting, we rely on the HMAR model [111]. This is applied on every detected bounding box of the input video and provides us with their initial, single frame, observations for 3D pose, appearance as well as location of the person in the 3D space.

As we link individual detections into tracklets, these representations are aggregated across each tracklet, allowing us to form temporal models, *i.e.*, functions for the aggregation and prediction of each representation separately (see right side of Fig. 2.2). More specifically, for appearance, we use the canonical UV map of the SMPL model to aggregate appearance, and employ its most recent prediction as person's appearance. For pose, we aggregate information using a modification of the HMMR model [64], where through its "movie strip" representation, we can produce 3D pose predictions. Finally, for 3D location, we use linear regression to predict the future location of the person.

This modeling enables us to develop our tracking system, **PHALP**(**P**redicting **H**uman **A**ppearance, **L**ocation and **P**ose for tracking), which aggregates information over time, uses it to predict future

states, and then associates the predictions with the detections. First, we predict the 3D location, 3D pose and 3D appearance for each tracklet for a short period of time. For a future frame, these predictions need to be associated with the detected people of the frame. To measure similarity, we adopt a probabilistic interpretation and compute the posterior probabilities of every detection belonging to each one of the tracklets, based on the three basic attributes. With the appropriate similarity metric, association is then easily resolved by means of the Hungarian algorithm. The newly linked detections can now update the temporal model of the corresponding tracklets for 3D appearance, 3D location and 3D pose in an online manner and we continue the procedure by rolling-out further prediction steps. The final output is an identity label for each detected bounding box in the video. Notably, this approach can also be applied on videos with shot changes, e.g., movies [51], with minor modifications. Effectively, we modify our similarity to include only appearance and 3D pose information for these transitions, since they (unlike location) are not affected by the shot boundary.

2.2 Related work

Tracking. Object tracking is studied in various settings such as single object tracking, multi-object tracking for humans, and multi-object tracking for vehicles etc. The tracking literature is vast and we refer readers to [28, 160, 25] for a comprehensive summary. In general, tracking can be applied on any category, however, in this section we discuss the methods that focus on tracking people. These approaches often work in a tracking-by-detection setting, where 2D location, 2D keypoint features [45, 125, 153] and 2D appearance [89, 156, 157] are used to associate detections over time. Quality of the detection plays a key role in this case and many works jointly learn or fine-tune their own detection models [10, 89]. In this work, we are interested in the effectiveness of 3D representations for tracking and thus assume that detected bounding boxes are provided, which we associate through our representations. On the other hand, tracking-by-regression [10, 39] predicts future locations using the knowledge of the past detections. While this alleviate the requirement for good quality detections, most of the works regress in the image plane. The projection from 3D world to the image plane makes it hard to make this prediction, therefore these methods need to learn non-linear motion models [2, 165, 10]. Compared to these methods, PHALP predicts short-term location in 3D coordinates, by simple linear regression. Additionally, we also predict appearance and pose features for better association.

Finally, there are methods that incorporate 3D information in tracking [88], however most of these approaches assume multiple input cameras [77, 167] or 3D point cloud observation from LiDAR [149]. In this chapter we focus on the setting where the input is a monocular video. Recent works track occluded people based on object permanence [68, 129]. These methods rely on depth estimation network to get 3D location [68] or need to learn permanence using sequence of annotated data [129]. However, by placing full human mesh in 3D space and predicting their location, pose and appearance, object permanence is already built into our system.

textbfMonocular 3D human reconstruction. Although there is a long history of monocular 3D human reconstruction methods, *e.g.*, [13, 52], here we focus on more recent works. Many of the relevant approaches rely on the SMPL model [83], which offers a low-dimensional parameterization

of the human body. HMR [63] has been one of the most notable ones, using a neural network to regress the SMPL body parameters from a single image. Follow-up works have improved the robustness [72, 75] and added additional features like estimation of camera parameters [73], or probabilistic estimation of pose [76]. Recently, Rajasegaran *et al*. [111] introduced HMAR, by adding an appearance head to HMR. Other works have focused on extending HMR to the temporal dimension, *e.g*., HMMR [64] and VIBE [71]. In this work, we make use of a modification of HMAR [111] as the main feature backbone, while employing a model similar to HMMR [64] for temporal pose prediction, but instead, using a transformer to aggregate pose information over time [99]. Regarding human motion prediction, Kanazawa *et al*. [64], regress future poses from the temporal pose representation of HMMR, the "movie-strip". Zhang *et al*. [163] extend this to PHD, employing autoregressive prediction of human motion. Aksan *et al*. [1] also regress future motion in an autoregressive manner, using a transformer.

2.3 Method

Tracking humans using 3D representations has significant advantages, including that 3D appearance is invariant to pose variations and 3D provides the ability to have amodal completion for humans during partial occlusion. Our tracking algorithm accumulates these 3D representations over time, to achieve better association with the detections. PHALP has three main stages: 1) estimating 3D representations for each human detection, 2) aggregating representations over time and predicting their future state, 3) associating tracklets with detections using predicted representations in a probabilistic framework. We explain each stage in the next sections.

Single-frame processing

The input to our system is a set of person detections along with their masks, estimated by conventional detection networks, like Mask-RCNN [53]. Each detection is processed by our backbone that computes the basic representations for pose, appearance and location on a single-frame basis. For this feature extraction we use a modification of the HMAR model [111]. HMAR returns a feature representation for the 3D pose $\mathbf{p}$, for appearance $\mathbf{a}$, while it can recover an estimate for the 3D location $\mathbf{l}$ of the person.

The HMAR model takes as input the pixels in the bounding box corresponding to a detected person. This means that in a crowded, multi-person scenario, the input can contain pixels from more than one person, potentially confusing the network. To deal with this problem, we modify HMAR to take as additional input, the pixel level mask of the person of interest (obtained from Mask R-CNN [53]) and re-train HMAR. Obviously, we cannot expect this step to be perfect, since there can be inaccuracies in the bounding box or mask estimates. However, we observed that the model is more robust in the case of close person-person interactions, which are common in natural videos.

3D tracklet prediction

The 3D estimates provide a rich and expressive representation for each detection. However, this is the result of single-frame processing. During tracking, as we expand each tracklet, we have access to more information that is representative of the state of the tracked person. To properly leverage this information, our tracking algorithm builds a tracklet representation after each step of the online processing, which allows us to predict the future states for each tracklet. In this section we describe how we build this tracklet representation, and more importantly, how we use it to *predict* the future state of each tracklet.

Appearance: The appearance pathway is used to integrate appearance information for each person over multiple frames. The single frame appearance representation for the person i at time step t, $\mathbf{A}_t^i$, is taken from the HMAR model by combining the UV image of that person $\mathbf{T}_t^i \in \mathcal{R}^{3\times256\times256}$ and the corresponding visibility map $\mathbf{V}_t^i \in \mathcal{R}^{1\times256\times256}$ at time step t:

$$\mathbf{A}_t^i = [\mathbf{T}_t^i, \mathbf{V}_t^i] \in \mathcal{R}^{4\times256\times256} \tag{2.1}$$

Note that the visibility mask $\mathbf{V}_t^i \in [0, 1]$ indicates whether a pixel in the UV image is visible or not, based on the Mask-RCNN mask. Now, if we assume that we have established the identity of this person in neighboring frames, we can integrate the partial appearance coming from the independent frames to an overall tracklet appearance for the person. Using the set of single frame appearance representations $\mathcal{A}^i = \{\mathbf{A}_t^i, \mathbf{A}_{t-1}^i, \mathbf{A}_{t-2}^i, ...\}$, after every new detection we create a singe per-tracklet appearance representation:

$$\widehat{\mathbf{A}}_t^i = \Phi_A(\widehat{\mathbf{A}}_{t-1}^i, \mathbf{A}_t^i) = (1-\alpha) * \widehat{\mathbf{A}}_{t-1}^i + \alpha * \mathbf{A}_t^i, \tag{2.2}$$

$$\text{where} \quad \alpha = \begin{cases} \alpha_0, & \text{if } \widehat{\mathbf{V}}_{t-1}^i = 1 \text{ and } \mathbf{V}_t^i = 1 \\ 1, & \text{if } \widehat{\mathbf{V}}_{t-1}^i = 0 \text{ and } \mathbf{V}_t^i = 1 \\ 0, & \text{if } \widehat{\mathbf{V}}_{t-1}^i = 1 \text{ and } \mathbf{V}_t^i = 0. \end{cases}$$

Here, Φ_A is the appearance aggregation function, which takes a weighted sum of the appearance representations from the previous tracklet and the new detection. Note that, at the start of the tracklet we simply assign the initial single-frame appearance to the tracklet ($\widehat{\mathbf{A}}_0^i = \mathbf{A}_0^i$). With this definition of Φ_A, we can aggregate appearance information over time, while allowing the representation to change slowly to account for slight appearance changes of the person during a video. Moreover, the UV image provides appearance of each point on the body surface independently of body pose and viewpoint which enables summation on the pixel space, without any learnable components. Figure 2.3 shows how the UV image of the person is aggregated over time and used for association of new detections.

For prediction, we make the realistic assumption that human appearance will not change rapidly over time. Then, the appearance of the tracklet $\widehat{\mathbf{A}}_t^i$ functions as a reasonable prediction for the future appearance of the person. Therefore, we use $\widehat{\mathbf{A}}_t^i$ as the prediction for appearance and use it to measure similarity against detections in future frames.

Location: Lifting humans from pixels into the 3D space allows us to place them in a 3D coordinate frame. Let us assume that a person i at time t has an estimated 3D location $\mathbf{l}_t^i \in \mathcal{R}^3$. Although, we

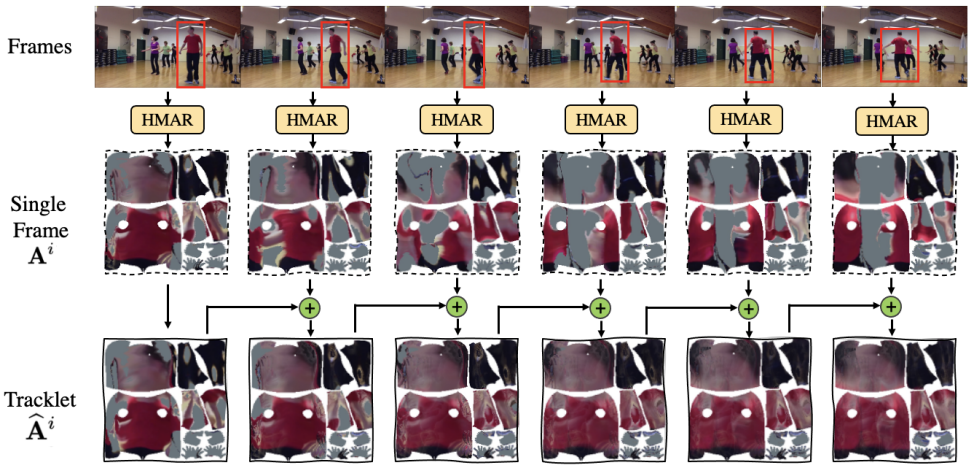


Figure 2.3: **Prediction of appearance:** The single frame appearance $\mathbf{A}^i$ is aggregated over time for the prediction of the tracklet appearance $\widehat{\mathbf{A}}^i$. At the start, only the front side of the *person* is visible, however as the person moves their visibility changes, making only their back side visible. For the single frame appearance, the visible regions change corresponding to the visibility of the person in the frame. However, for the tracklet, the appearance is accumulated over time. Even if the front side is not visible in the last frame, the tracklet can predict these regions using its past.

can get an estimate for the location of the person in the global camera frame, this tends to be noisy, particularly along the z-axis. To avoid any instabilities when it comes to predicting future location, instead of performing our prediction on the Euclidean $(X, Y, Z)^T$ space, we express our locations in an equivalent $\mathbf{l}_t^i = (x, y, n)^T$ space where (x, y) is the location of the root of the person in the pixel space and n is *nearness*, defined as log inverse depth $n = \log(1/z)$. Nearness is a natural parameterization of depth in vision settings, *e.g.*, [74], because of the $1/z$ scaling of perspective projection. In our case it corresponds to the scale of the human figures that we estimate directly from images. We independently linearly regress the location predictions for x, y and n. This is related to the Constant Velocity Assumption (CVA) used in past tracking literature [160], but there is a subtlety here because constant velocity in 3D need not give rise to constant velocity in 2D (a person would appear to speed up in the image as they approaches the camera). But local linearization is always a reasonable approximation to make, which is what we do.

Let us assume that a tracklet has a set of past locations $\mathcal{L}^i = \{\mathbf{l}_t^i, \mathbf{l}_{t-1}^i, \mathbf{l}_{t-2}^i, ...\}$. Then, the prediction of the location for time step $t + 1$ is given by:

$$\begin{aligned} \widehat{\mathbf{l}}_{t+1}^i &= (\widehat{x}_{t+1}^i, \widehat{y}_{t+1}^i, \widehat{n}_{t+1}^i)^T, \\ \text{where } \widehat{x}_{t+1}^i &= \Phi_L(\{x_t^i, x_{t-1}^i, x_{t-2}^i, ..., x_{t-w}^i\}, t+1). \end{aligned} \tag{2.3}$$

Here, Φ_L is the location aggregation function and we use a simple linear regression for prediction in our tracking algorithm. We predict $\widehat{y}_{t+1}^i$ and $\widehat{n}_{t+1}^i$ in a similar fashion. Φ_L takes the last w observations to fit a line by least squares and regress the future location for x, y and n independently. From the theory of linear regression, the prediction interval for x at a time step t' is given by the

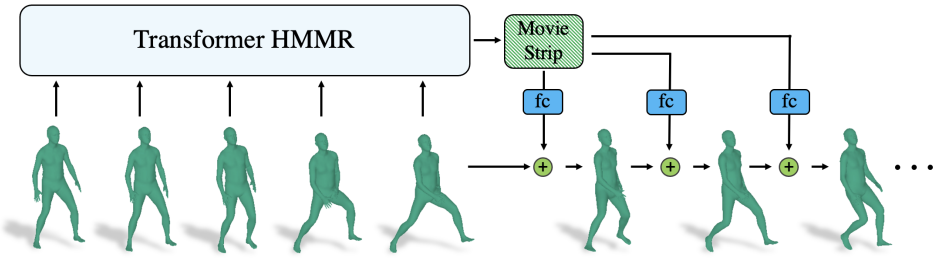


Figure 2.4: **Prediction of Pose:** We use a modified version of HMMR [64] with transformer backbone. Having transformer as the backbone gives us the flexibility to have missing people in the tracklet (by masking the attention [99]), while still allowing us to predictions of future poses. Finally, the transformer gives us a movie-strip representation and that is used to regress future poses.

equation below:

$$\delta_x(t') = t_{(1-\alpha/2)} \times \sqrt{MSE \times \left(1 + \frac{1}{w} + \frac{(t'-\bar{t})^2}{\sum(t-\bar{t})^2}\right)}. \tag{2.4}$$

Here, $t_{(1-\alpha/2)}$ is the Student's t distribution with confidence α and degree of freedom $w-2$. MSE is the mean squared error on the predicted locations and $\bar{t}$ is the mean of the time stamps for the previous observations. We similarly compute prediction intervals Δ_y, Δ_n for y, n respectively.

Pose: For the pose pathway, we integrates pose information across the tracklet and predicts poses for the near future. To do this, we follow the HMMR architecture [64]. Effectively, we learn a function Φ_P that takes as input a series of pose embeddings of a person $\mathcal{P}^i = \{\mathbf{p}^i_t, \mathbf{p}^i_{t-1}, \mathbf{p}^i_{t-2}, ...\}$ and computes a temporal pose embedding $\widehat{\mathbf{p}}_t$. We train this temporal pose aggregation function Φ_P to smooth the pose $\widehat{\mathbf{p}}^i_t$ at frame t, and regress future pose representations $\{\widehat{\mathbf{p}}^i_{t+1}, \widehat{\mathbf{p}}^i_{t+2}, ..., \widehat{\mathbf{p}}^i_{t+c}\}$ (for up to $c = 12$ frames in the future). We use a transformer to compute Φ_P [99]. This choice allows for additional flexibility, since there are frames where the identity cannot be detected (*e.g.*, due to occlusions). Transformer can handle this scenario gracefully, by not attending to these frames.

Tracking with predicted 3D representations

Given the bounding boxes and their single-frame 3D representations, our tracking algorithm associates identities across frames in an online manner. At every frame, we make future predictions for each tracklet and we measure the similarity with the detected single-frame representation. More specifically, let us assume that we have a tracklet T_i, which has been tracked for a sequence of frames and has information for appearance, pose and location. The tracklet predicts its appearance $\widehat{\mathbf{A}}$, pose $\widehat{\mathbf{p}}$ and location $\widehat{\mathbf{l}}$ for the next frame, and we need to measure a similarity score between these predictions of the tracklet T_i and a detection D_j to make an association. Our tracklet representation has three different attributes (appearance, location and pose), so, directly combining their similarities/distances would not be ideal, since, each attribute has different characteristics. Instead, we investigate the conditional distributions of inliers and outliers of the attributes. Figure 2.5

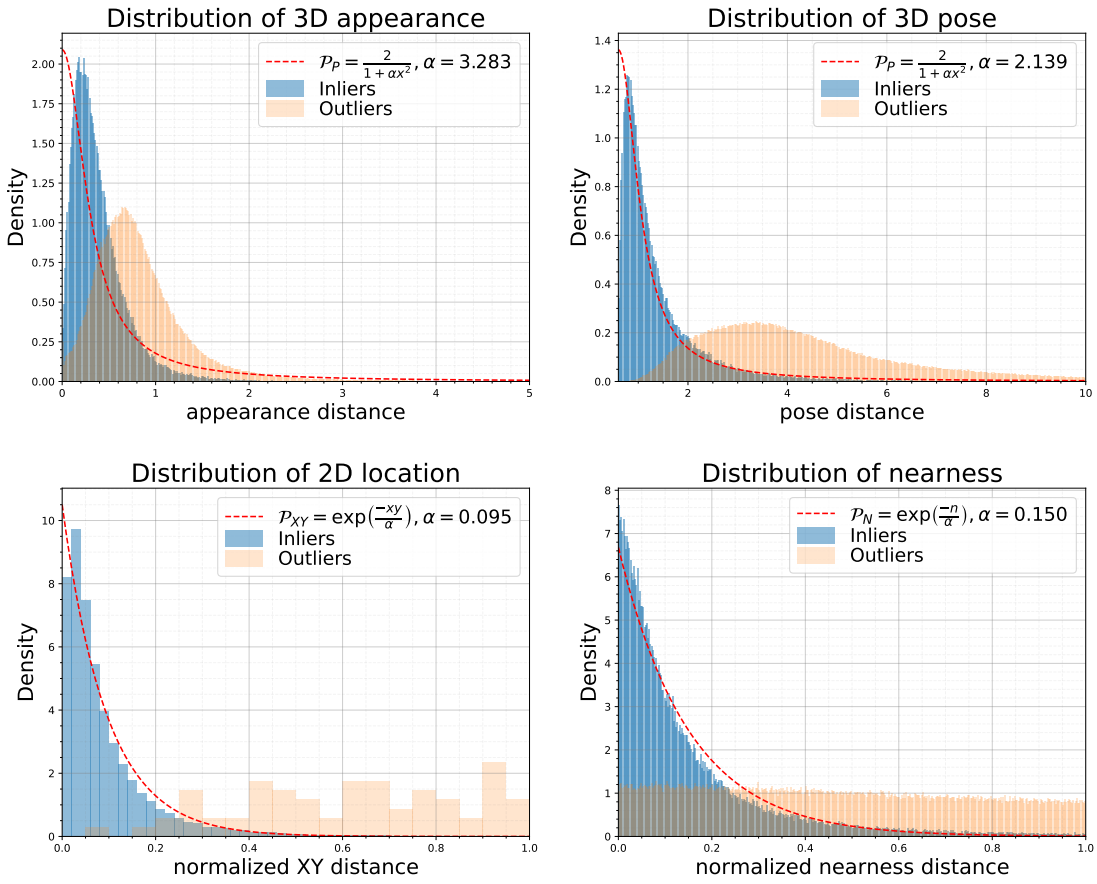


Figure 2.5: **Conditional distributions of the attribute distances:** We plot the data for the distances between the tracklet prediction and the single frame detection using annotated data from PoseTrack [3]. The curves show the distributions of correct matches (inliers) and incorrect matches (outliers). For 2D location and nearness the distances are normalized by the prediction interval.

presents the corresponding probability distributions for the PoseTrack dataset [3]. The characteristics of these distributions motivate our design decisions for our further modeling.

Assume that tracklet T_i has an appearance representation $\widehat{\mathbf{A}}_t^i$. On the detection side, the detection D_j has a single-frame appearance representation $\mathbf{A}_t^j$. Both of these representations are in the UV pixel space, therefore we first encode them into an embedding space using the HMAR appearance-encoder network. This gives us an appearance embedding $\widehat{\mathbf{a}}_t^i$ and $\mathbf{a}_t^j$ for the prediction of the tracklet T_i and detection D_j, respectively. We are interested in estimating the posterior probability of the event where the detection D_j belongs to the tracklet T_i, given some distance measure of the appearance feature (Δ_a). Assuming that the appearance distance is $\Delta_a = ||\widehat{\mathbf{a}}_t^i - \mathbf{a}_t^j||_2^2$, then the posterior probability is proportional to the conditional probability of the appearance distances, given correct assignments based on Bayes rule. We model this conditional probability as a Cauchy

distribution, based on the observations from the inlier distribution of appearance distances (see also Fig 2.5):

$$\mathcal{P}_A(D_j \in T_i | d_a = \Delta_a) \propto \frac{1}{1 + \beta_a \Delta_a} \tag{2.5}$$

The distribution has one scaling hyper-parameter β_a.

Similarly, for pose, we use Cauchy distribution to model the conditional probability of inlier distances. We measure pose distance $\Delta_p = ||\widehat{\mathbf{p}}_t^i - \mathbf{p}_t^j||_2^2$ between the predicted pose representation $\mathbf{p}_t^i$ from the tracklet T_i and the pose representation $\mathbf{p}_t^j$ of detection D_j. The posterior probability that the detection belongs to the track, given the pose distance is:

$$\mathcal{P}_P(D_j \in T_i | d_p = \Delta_p) \propto \frac{1}{1 + \beta_p \Delta_p} \tag{2.6}$$

Here, $\Delta_p = ||\widehat{\mathbf{p}}_t^i - \mathbf{p}_t^j||_2^2$ and β_p is the scaling factor.

For location, let us assume the tracklet T_i has predicted a location $\widehat{\mathbf{l}}_t^i = (\widehat{x}_t^i, \widehat{y}_t^i, \widehat{n}_t^i)^T$ with a set of prediction intervals $\{\delta_x, \delta_y, \delta_n\}$, and the detection D_j is at a 3D location $\mathbf{l}_t^j = (x_t^j, y_t^j, n_t^j)^T$. We treat the 3D coordinates x, y and the nearness term n coordinates independently, and compute the posterior probabilities of the detection belongs to the tracklet given the location distance. We model the conditional probability distribution as an exponential distribution, based on the findings from the empirical data. The Fig 2.5 shows the distribution of 2D distance and nearness distance, scaled by the confidence interval, of inliers approximately follow the exponential distribution:

$$\mathcal{P}_{XY}(D_j \in T_i | d_{xy} = \Delta_{xy}) \propto \frac{1}{\beta_{xy}} \exp\left(\frac{-\Delta_{xy}}{\beta_{xy}\delta_{xy}}\right). \tag{2.7}$$

Here, β_{xy} is a scaling parameter for the exponential distribution, Δ_{xy} is the 2D pixel distance between the predicted tracklet and the detection and $\delta_{xy} = \sqrt{\delta_x^2 + \delta_y^2}$ is the prediction interval for the 2D location prediction. The expression for the posterior probability for nearness $\mathcal{P}_N$ is similar:

$$\mathcal{P}_N(D_j \in T_i | d_n = \Delta_n) \propto \frac{1}{\beta_n} \exp\left(\frac{-\Delta_n}{\beta_n \delta_n}\right). \tag{2.8}$$

Here, β_n is the scaling parameter for the exponential distribution, δ_n is the confidence interval for the nearness prediction, and Δ_n is the L_1 distance between the nearness of the tracklet prediction and the detection.

Given the conditional probabilities of the detection belonging to a tracklet conditioned on the individual cues (appearance, location, pose), we can compute the overall conditional probability of the detection D_j belonging to the tracklet T_i, given all the cues (by making the simplifying assumption that they are independent:

$$\mathcal{P}(D_j \in T_i | \Delta_a, \Delta_p, \Delta_{xy}, \Delta_n) \propto \mathcal{P}_A \mathcal{P}_P \mathcal{P}_{XY} \mathcal{P}_N. \tag{2.9}$$

This allow us to estimate how probable an association is based on various attribute distances. Finally, we map the similarity measures (probability values up to a scale), to cost values, for solving

Algorithm 1 Tracking Algorithm

procedure PHALP TRACKING
Require: All active tracklets $\mathcal{T}$, all detections and their single frame 3D representations at time t, $\mathcal{D}$ and maximum age of a tracklet t_{max}.
  for $T_j \in \mathcal{T}$ **do**
    # Predict all attributes for the next frame.
    $\widehat{\mathbf{A}}_t^j \leftarrow \Phi_A(\{\mathbf{A}_{t-1}^j, \mathbf{A}_{t-2}^j, ...\})$
    $\widehat{\mathbf{p}}_t^j \leftarrow \Phi_P(\mathbf{p}_{t-1}^j, \mathbf{p}_{t-1}^j, ...\})$
    $\widehat{\mathbf{l}}_t^j \leftarrow \Phi_L(\{\mathbf{l}_{t-1}^j, \mathbf{l}_{t-2}^j, ...\})$
  # Compute cost between tracklets and detections.
  $\mathbf{C}_{i,j} \leftarrow \Phi_C(D_i, T_j)$ for all $D_i \in \mathcal{D}$ and $T_j \in \mathcal{T}$
  # Hungarian to assign detections to tracklets.
  $\mathcal{M}, \mathcal{T}_u, \mathcal{D}_u \leftarrow \texttt{Assignment}(\mathbf{C})$
  # Update the matched tracklets.
  $\mathcal{T} \leftarrow \{T_j(D_i), \ \forall (i,j) \in \mathcal{M}\}$
  # Increase the age of unmatched tracletks.
  $\mathcal{T} \leftarrow \{T_j(age) += 1, \ \forall (j) \in \mathcal{T}_u\}$
  # Make new tracklets with unmatched detections.
  $\mathcal{T} \leftarrow \{T_j(D_i), \ \forall (i) \in \mathcal{D}_u, \ j = |\mathcal{T} + 1|\}$
  Kill the tracklets with age $\geq t_{max}$.
return Tracklets $\mathcal{T}$

association. The cost function between the detection representations and a predicted representations of the tracklet is defined as:

$$\begin{aligned} \Phi_{C(D_j,T_i)} &= -\log(\mathcal{P}(D_j \in T_i)) \\ &= -\log(\mathcal{P}_A) - \log(\mathcal{P}_P) - \log(\mathcal{P}_{XY}) - \log(\mathcal{P}_N), \end{aligned} \tag{2.10}$$

where the second equality is up to an additive constant. Once the cost between all the tracklets and the detection is computed, the Hungarian algorithm solves the association.

Estimating the parameters of the cost function: The cost function Φ_C has 4 parameters ($\beta_a, \beta_p, \beta_{xy}$ and β_n). Additionally, the Hungarian algorithm has one parameter β_{th} to decide whether the tracklet is not a match to the detection. Therefore, overall we have five parameters for the association part of our method. We treat this as an empirical risk minimization problem and optimize the β values based on a loss function. We initialize $\beta_a, \beta_p, \beta_{xy}$ and β_n with the values from the estimated density functions and use frame level association error as a loss function for the optimization. We use the Nelder–Mead algorithm [93] for this optimization. Finally, the optimized β values are used for the cost function across all datasets.

Extension to shot changes

Our framework can be easily extended to also handle shot changes, which are common in edited media (movies, TV shows, sports, etc). We use a shot detector [58] to reliably identify frames that indicate shot changes. Knowing the shot boundary, during tracking, we update the distance metric accordingly. More specifically, since appearance and 3D pose are invariant to the viewpoint,

we keep these factors in the distance computation, while we drop the location distance from the distance metric, because of the change in the camera location. Then, the association is computed based on this updated metric. We demonstrate this utility on the AVA dataset [51] and present results in Section 2.4.

2.4 Experiments

In this section, we present the experimental evaluation of our approach. We report results on three datasets: PoseTrack [3], MuPoTS [87] and AVA [51], which capture a diverse set of sequences, including sports, casual interactions and movies. Our method operates on detections and masks coming from an off-the-shelf Mask-RCNN network [53], and returns the identity label for each one of them. Therefore, the metrics we use to report results also focus on identity tracking at the level of the bounding box. More specifically, we report results using Identity switches (IDs), Multi-Object Tracking Accuracy (MOTA) [65], ID F1 score (IDF1) [115] and HOTA [86]. In all cases, we adopt the protocols of Rajasegaran *et al*. [111] for evaluation.

First, we ablate the main components of our approach. Specifically, we investigate the effect of each one of the tracking cues we employ, *i.e*., 3D appearance, 3D location and 3D pose, and how they affect the overall tracking pipeline. For this comparison, we report results on the PoseTrack dataset [3]. The full results are presented in Table 2.1. As we can see, removing each one of the main cues leads to degradation in the performance of the system, where 3D location seems to have the largest effect on the performance, followed by appearance and 3D pose. While the effect of 3D pose is not significant as other cues, computing the 3D pose is essential because a) it allows us to get a good location estimate, and b) it guides the appearance calculation. Moreover, this ablation also highlights the importance of having the nearness term in the cost function, a feature that is not available to purely 2D tracking methods. Additionally, we evaluate our system when only the 3D location is included in the cost function. This has a negative effect on the tracking performance, validating the importance of the other cues. Finally, we evaluate our method without masks, during the appearance aggregation and HMR stage. Our experiment shows that access to masks improves performance.

Next, we evaluate our approach in comparison with the state-of-the-art methods. The results are presented in Table 2.2. We report results on PoseTrack [3], MuPoTS [87] and AVA [51]. Our method outperforms the previous baselines, as well as the state-of-the-art approach of Rajasegaran [111]. The gains are significant across all metrics. Our method also outperforms the other approaches in the HOTA metric.

Finally, we provide qualitative results of our method in Fig 2.6. These results indicate that our method performs reliably even in very hard occlusion cases, while it can recover the correct identity over multiple successive occlusions. Please note the robustness of our method in complex motion sequences, shot changes and long trajectories. For visualization we use a colored head-mask to represent a unique track.

More design choices: PHALP has a modular design, where each components can be easily replaced. We demonstrate this by altering some of our original choices. For example, we replace

Figure 2.6: **Qualitative Results:** We show the tracking performance of PHALP in various datasets (frame number is shown at the top left corner). While all the tracks have full 3D reconstructions, we only use the **head-masks** to visualize the tracks since it is easier to spot errors. The first three rows are from the PoseTrack dataset [3]. These results show that even during successive occlusions PHALP is able to track the identity of the correct person. Note that, in the first row, although the two persons with the green head-mask and the purple head-mask have similar appearance, our method can track each one of them successfully. In the second row, the player is going through multiple occlusions, yet recovered correctly. The third row shows the robustness of our linearization approximation for 3D location prediction, even when the motions of the players are very complex. In the MuPoTS dataset [87] (4th row), our method can handle very close interactions between people. This is due to the fact that, our modification of HMAR recovers meshes conditioned on the detected mask. We also show results (5th row) on the AVA dataset [51]. After the 3rd frame, there is a shot change in the video, and the person with the yellow head-mask is tracked successfully across the shots. Finally, we show qualitative results on a MOT17 sequence. The person with the blue head-mask is tracked for the whole sequence while they are going through multiple occlusions for a long time. More results at the PHALP website.

Method	PoseTrack		
	IDs↓	MOTA↑	IDF1↑
w/o 3D appearance	632	58.4	74.9
w/o 3D pose	558	58.9	76.2
w/o location	948	57.3	71.6
w/o nearness	622	58.5	74.8
w/o pose, appearance	645	57.5	74.7
w/o mask	573	58.1	75.3
Full system	**541**	**58.9**	**76.4**

Table 2.1: **Ablation of the main components of PHALP on PoseTrack [3].** Removing each tracking cue (3D appearance, 3D pose or 3D location) leads to degradation in the performance.

Method	Posetrack				MuPoTS				AVA	
	IDs↓	MOTA↑	IDF1↑	HOTA↑	IDs↓	MOTA↑	IDF1↑	HOTA↑	IDs↓	IDF1↑
Trackformer [89]	1263	33.7	64.0	46.7	43	24.9	62.7	53.2	716	40.9
Tracktor [10]	702	42.4	65.2	38.5	53	51.5	70.9	50.3	289	46.8
AlphaPose [37]	2220	36.9	66.9	37.6	117	37.8	67.6	41.8	939	41.9
FlowPose [155]	1047	15.4	64.2	38.0	49	21.4	67.1	43.0	452	52.9
T3DP [111]	655	55.8	73.4	50.6	38	62.1	79.1	59.2	240	61.3
PHALP	**541**	**58.9**	**76.4**	**52.9**	**22**	**66.2**	**81.4**	**59.4**	**227**	**62.7**

Table 2.2: **Comparison with state-of-the-art tracking methods.** We compare our method, PHALP, with various tracking methods in three different datasets. Our approach outperforms the other baselines across all datasets and metrics.

MaskRCNN [53] with PointRend [69] at the detection stage. However, we did not observe any significant improvement on tracking metrics with this change. Moreover, we replace the inferred 3D location from HMAR with the equivalent location inferred by PARE [72]. This change improves overall tracking performance by 5% in terms of ID switches. Finally, we also investigate the option of postponing the final decision about the identity of a detection. We refer to this as LMC (Lookback Merge Check), which can connect tracklets that have been over-segmented. With LMC, for every new tracklet, we regress backwards (*i.e.*, past location) and check if there is any older tracklet in that location with similar appearance to the new tracklet. If such a tracklet exists, then we merge these two tracklets. This lookback merge and check (LMC) can help to achieve roughly 20 less ID switches in the PoseTrack dataset. For more implementation details and for detailed results for all these experiments, please see the SupMat.

2.5 Discussion

We presented PHALP, an approach for monocular people tracking, by predicting appearance, location and pose in 3D. Our method relies on a powerful backbone for 3D human mesh recovery, modeling on the tracklet level for collecting information across a tracklet's trajectory, and eventually predicting the future states of each tracklet. One of the main benefits of PHALP is that the association aspect requires tuning of only five parameters, which makes it very friendly for training on small-scale tracking datasets, where annotating the identity of every person in a video can be expensive. Our approach can be naturally extended to make use of more attributes, *e.g.*, a face embedding, which could be useful for cases with close-ups, like movies.

The main assumptions for PHALP are that we have access to a good object detector for the initial bounding box/mask detection, and a strong HMAR network for single-frame lifting of people to 3D. If the performance of these components is not satisfactory, it can also affect tracking quality. Regarding societal impact, tracking systems can be used to monitor patients or help with some treatments [27]. On the other hand, tracking systems have often been used for human surveillance. We do not condone such use. Instead, we believe that a tracking system will be valuable for studying social-human interactions.

Acknowledgements: This work was supported by ONR MURI (N00014-14-1-0671), the DARPA Machine Common Sense program, as well as BAIR and BDD sponsors.

2.6 Additional Details

We include more implementations details about our approach. We provide more details about the experiments for this chapter (Sections 2.8). Finally, we extend the discussion about the failure cases of our system (Sections 2.9). Additionally, we encourage the readers to also watch the attached supplementary video, which is also available here: `https://brjathu.github.io/PHALP`.

2.7 Implementation details

Architecture: First, we provide some additional architectural details about the networks used in our pipeline. Regarding the HMR module, the architecture is similar to [99]. For the mask conditioning, we use the detections and masks from MaskRCNN [53] as a masking operation to mask out features that do not belong to the person of interest. This masking operation does not require any extra parameters, and it only acts on the last feature map of the convolutional part of ResNet. For the appearance head of HMAR, we use the same design as [111]. The only difference is that for the texture encoder, the input has four channels (RGB & mask), where the mask is used to indicate locations on the body that have not been visible during the video, thus invalid. Finally, for the HMMR part, we use a transformer similar to [99], with one layer and one head. The functionality is similar to the original HMMR [64], but for the future poses, instead of regressing a residual on the parameter space (θ, β), the residual is on the feature space.

Training: Next, we provide more details of the training procedure. First we train the HMR model, followed by the appearance head of HMAR and then the temporal head of HMMR model. For the HMR model, we follow the training details of [99], with the additional modification of using the mask conditioning part. Once, the HMR model is trained, we use it as the backbone for the training of HMAR model. For HMAR, we apply the estimated texture on the SMPL body and project it to the image. Then, we optimize the loss such that the texture of the projection matches the actual RGB values *on the segmented (MaskRCNN) pixels*. We train this model for 10000 iterations with an Adam optimizer with an initial learning rate of 0.001. HMAR is trained on a per frame basis. Finally, for the HMMR model, we use data from Human3.6M [59] and InstaVariety [64]. We train the HMMR head for 10000 iterations with a learning rate of 0.0001. HMR, HMAR and HMMR require training times of about 5, 3 and 0.2 days, respectively on a single NVIDIA 2080 Ti GPU. At inference, PHALP can run at about 7 FPS, however with an optimized code it can run much faster.
LMC: We observe that the many failures in our tracking system is caused by missing detections or out-of-distribution poses in the videos. While, solving the detection or human pose reconstruction is not the scope of this chapter, we propose a simple solution to overcome these failures at the tracking stage. For each new tracks, we wait for 7 frames for this track accumulated enough information. Then, we take these 7 detected locations and their corresponding time-steps to regress k frames back into the past. The value k is determined by the old tracks which have been not updated for $k + 7$ frames. Once we have a predicted location of the new track into the past we measure 3D location distance between old tracks and predicted past location. In addition to the location we also measure the similarity between the appearance of the old and new tracks, assuming the appearance does not change over time. Finally, our cost function for LMC involves location distance and appearance distance and we solve it via Hungarian to assign new tracks to old tracks.

2.8 Experimental details

For the evaluation on PoseTrack [3], MuPoTS [88] and AVA [51], we follow the test protocols of Rajasegaran *et al*. [111]. For evaulating these methods, we only reject the detections if the IOU distance is zero. However, the non-rejected detections will have to compete for the ground-truth via a Hungarian matching algorithm. This is to avoid penalizing the methods based on their quality of detections.

Additionally, we also evaluate the robustness of appearance aggregation, by adding pixel noise to the visibility masks Eq 2.2. Adding noise even to 90% of the pixels increases the IDs metric on PoseTrack only by 4%, meaning that appearance aggregation is quite robust. We are not so prone to drifting due of occlusions, because the appearance of a body part will not update when this body part is not visible. We observe that we are robust to the value choice of α_0, with only $\pm 2\%$ change in the IDs metric on Posetrack for values of $\alpha_0 \in [0.1, 0.9]$.

Finally, we also test the effect of number of people in a video against the ID switches. Interestingly, we observe that, the errors scale almost linearly with the number of people. This suggests that PHALP can work on crowed scenes.

Figure 2.7: *LMC results:* We show to failure cases in our method. In the first example, MaskRCNN fails to detect the person (pointed in red) for long period of time. This causes the error in location prediction and therefore when this person detected again a new track is created. In the second example, HMR fails when the human pose is very different and causes a large cost in the pose distance. Due to this a new track is created for the same person. With LMC, we are able to look back and check whether these new track can be connected with any old tracks, and we connected them together as a single track.

Method	PoseTrack		
	IDs↓	MOTA↑	IDF1↑
PointRend Mask	558	58.9	76.2
PARE [72] Location	512	58.8	76.3
PHALP+LMC	520	58.9	76.3

Table 2.3: **Ablation of different design choices for PHALP.** Due to the modular architecture of PHALP, we can replace different components of it at different stages. We evaluate PHALP with replacing MaskRCNN with PointRend, and HMR location with PARE [72] location. We also evaluate PHALP+LMC, where we allow new tracks to connect with old tracks. This flexibility allows us to overcome the mistakes made at the detection stage and HMR stage.

2.9 Failure cases

Our method PHALP relies on 1) MaskRCNN masks and 2) HMAR pose. Most of our failure cases can be attributed to mistakes in these two methods. For example, non-maximum suppression for Mask RCNN is imperfect. This will create two detections for a single person or give a joint mask of two people. These masks, will hurt pose estimation, and then the location of the person followed by

bad appearance representation. This will eventually affect the tracking performance too. On the other hand, for challenging cases, HMR can also give bad SMPL reconstructions. For example, when a person is heavily occluded, the number of visible pixels is very small and this will affect the pose prediction of HMR. Although PHALP depends on these two methods, the robust line fitting, averaging the appearance and pose smoothing with the HMMR model can recover from occasional failures of these methods.

Chapter 3

On the Benefits of 3D Tracking and Pose for Human Action Recognition

In this work we study the benefits of using tracking and 3D poses for action recognition. To achieve this, we take the Lagrangian view on analysing actions over a trajectory of human motion rather than at a fixed point in space. Taking this stand allows us to use the tracklets of people to predict their actions. In this spirit, first we show the benefits of using 3D pose to infer actions, and study person-person interactions. Subsequently, we propose a Lagrangian Action Recognition model by fusing 3D pose and contextualized appearance over tracklets. To this end, our method achieves state-of-the-art performance on the AVA v2.2 dataset on both pose only settings and on standard benchmark settings. When reasoning about the action using only pose cues, our pose model achieves **+10.0 mAP** gain over the corresponding state-of-the-art while our fused model has a gain of **+2.8 mAP** over the best state-of-the-art model. Code and results are available at: https://brjathu.github.io/LART

3.1 Introduction

In fluid mechanics, it is traditional to distinguish between the Lagrangian and Eulerian specifications of the flow field. Quoting the Wikipedia entry, *"Lagrangian specification of the flow field is a way of looking at fluid motion where the observer follows an individual fluid parcel as it moves through space and time. Plotting the position of an individual parcel through time gives the pathline of the parcel. This can be visualized as sitting in a boat and drifting down a river. The Eulerian specification of the flow field is a way of looking at fluid motion that focuses on specific locations in the space through which the fluid flows as time passes. This can be visualized by sitting on the bank of a river and watching the water pass the fixed location."*

These concepts are very relevant to how we analyze videos of human activity. In the Eulerian viewpoint, we would focus on feature vectors at particular locations, either (x, y) or (x, y, z), and consider evolution over time while staying fixed in space at the location. In the Lagrangian viewpoint, we would track, say a person over space-time and track the associated feature vector across space-time.

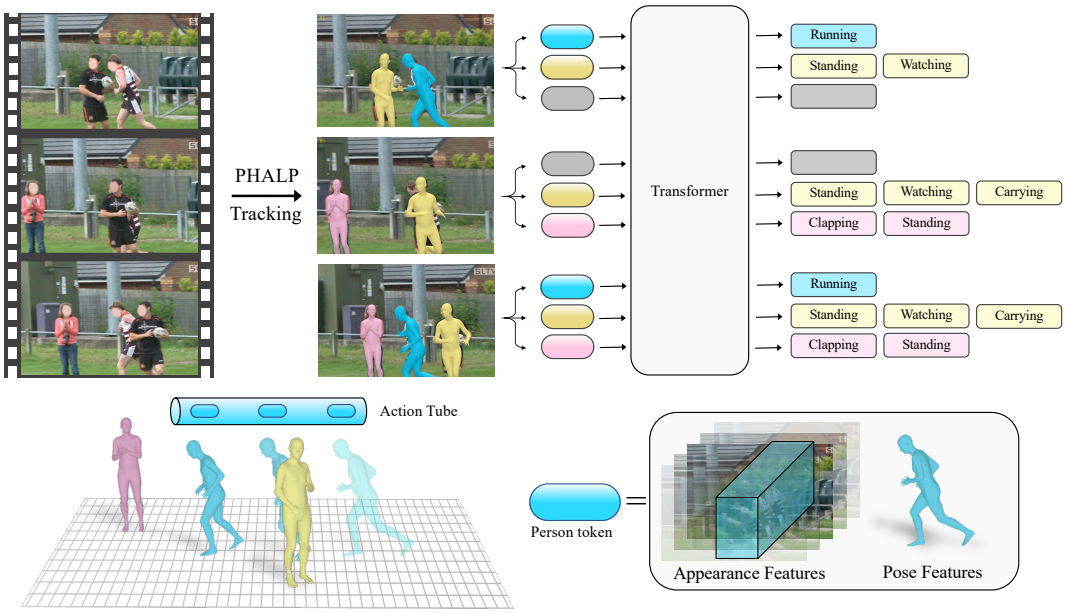


Figure 3.1: **Overview of our method:** Given a video, first, we track every person using a tracking algorithm (*e.g.* PHALP [110]). Then every detection in the track is tokenized to represent a human-centric vector (*e.g. pose, appearance*). To represent 3D pose we use SMPL [83] parameters and estimated 3D location of the person, for contextualized appearance we use MViT [35] (pre-trained on MaskFeat [146]) features. Then we train a transformer network to predict actions using the tracks. Note that, at the second frame we do not have detection for the blue person, at these places we pass a mask token to in-fill the missing detections.

While the older literature for activity recognition *e.g.*, [33, 138, 46] typically adopted the Lagrangian viewpoint, ever since the advent of neural networks based on 3D space-time convolution, *e.g.*, [132], the Eulerian viewpoint became standard in state-of-the-art approaches such as SlowFast Networks [41]. Even after the switch to transformer architectures [136, 35] the Eulerian viewpoint has persisted. This is noteworthy because the tokenization step for transformers gives us an opportunity to freshly examine the question, *"What should be the counterparts of words in video analysis?"*. Dosovitskiy *et al.* [32] suggested that image patches were a good choice, and the continuation of that idea to video suggests that spatiotemporal cuboids would work for video as well.

On the contrary, in this work we take the Lagrangian viewpoint for analysing human actions. This specifies that we reason about the *trajectory of an entity* over time. Here, the entity can be low-level, *e.g.*, a pixel or a patch, or high-level, *e.g.*, a person. Since, we are interested in understanding human actions, we choose to operate on the level of *"humans-as-entities"*. To this end, we develop a method that processes trajectories of people in video and uses them to recognize their action. We recover these trajectories by capitalizing on a recently introduced 3D tracking method PHALP [110] and HMR 2.0 [47]. As shown in Figure 3.1 PHALP recovers person tracklets

from video by lifting people to 3D, which means that we can both link people over a series of frames and get access to their 3D representation. Given these 3D representations of people (*i.e.*, 3D pose and 3D location), we use them as the basic content of each token. This allows us to build a flexible system where the model, here a transformer, takes as input tokens corresponding to the different people with access to their identity, 3D pose and 3D location. Having 3D location of the people in the scene allow us to learn interaction among people. Our model relying on this tokenization can benefit from 3D tracking and pose, and outperforms previous baseline that only have access to pose information [24, 119].

While the change in human pose over time is a strong signal, some actions require more contextual information about the appearance and the scene. Therefore, it is important to also fuse pose with appearance information from humans and the scene, coming directly from pixels. To achieve this, we also use the state-of-the-art models for action recognition [35, 81] to provide complementary information from the contextualized appearance of the humans and the scene in a Lagrangian framework. Specifically, we densely run such models over the trajectory of each tracklet and record the contextualized appearance features localized around the tracklet. As a result, our tokens include explicit information about the 3D pose of the people and densely sampled appearance information from the pixels, processed by action recognition backbones [35]. Our complete system outperforms the previous state of the art by a large margin of **2.8 mAP**, on the challenging AVA v2.2 dataset.

Overall, our main contribution is introducing an approach that highlights the effects of tracking and 3D poses for human action understanding. To this end, in this work, we propose a **L**agrangian **A**ction **R**ecognition with **T**racking (**LART**) approach, which utilizes the tracklets of people to predict their action. Our baseline version leverages tracklet trajectories and 3D pose representations of the people in the video to outperform previous baselines utilizing pose information. Moreover, we demonstrate that the proposed Lagrangian viewpoint of action recognition can be easily combined with traditional baselines that rely only on appearance and context from the video, achieving significant gains compared to the dominant paradigm.

3.2 Related Work

Recovering humans in 3D: A lot of the related work has been using the SMPL human body model [83] for recovering 3D humans from images. Initially, the related methods were relying on optimization-based approaches, like SMPLify [12], but since the introduction of the HMR [63], there has been a lot of interest in approaches that can directly regress SMPL parameters [83] given the corresponding image of the person as input. Many follow-up works have improved upon the original model, estimating more accurate pose [75] or shape [23], increasing the robustness of the model [99], incorporating side information [73, 76], investigating different architecture choices [72, 162], etc.

While these works have been improving the basic single-frame reconstruction performance, there have been parallel efforts toward the temporal reconstruction of humans from video input. The HMMR model [64] uses a convolutional temporal encoder on HMR image features [63] to

reconstruct humans over time. Other approaches have investigated recurrent [71] or transformer [99] encoders. Instead of performing the temporal pooling on image features, recent work has been using the SMPL parameters directly for the temporal encoding [9, 114].

One assumption of the temporal methods in the above category is that they have access to tracklets of people in the video. This means that they rely on tracking methods, most of which operate on the 2D domain [10, 37, 90, 155] and are responsible for introducing many errors. To overcome this limitation, recent work [111, 110] has capitalized on the advances of 3D human recovery to perform more robust identity tracking from video. More specifically, the PHALP method of Rajasegaran *et al*. [110] allows for robust tracking in a variety of settings, including in the wild videos and movies. Here, we make use of the PHALP system to discover long tracklets from large-scale video datasets. This allows us to train our method for recognizing actions from 3D pose input.

Action Recognition: Earlier works on action recognition relied on hand-crafted features such as HOG3D [70], Cuboids [31] and Dense Trajectories [138, 137]. After the introduction of deep learning, 3D convolutional networks became the main backbone for action recognition [128, 132, 18]. However, the 3D convolutional models treat both space and time in a similar fashion, so to overcome this issue, two-stream architectures were proposed [123]. In two-steam networks, one pathway is dedicated to motion features, usually taking optical flow as input. This requirement of computing optical flow makes it hard to learn these models in an end-to-end manner. On the other hand, SlowFast networks [41] only use video streams but at different frame rates, allowing it to learn motion features from the fast pathway and lateral connections to fuse spatial and temporal information. Recently, with the advancements in transformer architectures, there has been a lot of work on action recognition using transformer backbones [92, 11, 4, 35].

While the above-mentioned works mainly focus on the model architectures for action recognition, another line of work investigates more fine-grained relationships between actors and objects [142, 141, 127, 166]. Non-local networks [142] use self-attention to reason about entities in the video and learn long-range relationships. ACAR [97] models actor-context-actor relationships by first extracting actor-context features through pooling in bounding box region and then learning higher-level relationships between actors. Compared to ACAR, our method does not explicitly design any priors about actor relationships, except their track identity.

Along these lines, some works use the human pose to understand the action [24, 158, 147, 119, 135]. PoTion [24] uses a keypoint-based pose representation by colorizing the temporal dependencies. Recently, JMRN [119] proposed a joint-motion re-weighting network to learn joint trajectories separately and then fuse this information to reason about inter-joint motion. While these works rely on 2D key points and design-specific architectures to encode the representation, we use more explicit 3D SMPL parameters.

3.3 Method

Understanding human action requires interpreting multiple sources of information [67]. These include head and gaze direction, human body pose and dynamics, interactions with objects or other humans or animals, the scene as a whole, the activity context (e.g. immediately preceding actions by self or others), and more. Some actions can be recognized by pose and pose dynamics alone, as demonstrated by Johansson *et al* [61] who showed that people are remarkable at recognizing *walking*, *running*, *crawling* just by looking at moving point-lights. However, interpreting complex actions requires reasoning with multiple sources of information e.g. to recognize that someone is slicing a tomato with a knife, it helps to see the knife and the tomato.

There are many design choices that can be made here. Should one use "disentangled" representations, with elements such as pose, interacted objects, etc, represented explicitly in a modular way? Or should one just input video pixels into a large capacity neural network model and rely on it to figure out what is discriminatively useful? In this chapter, we study two options: a) human pose reconstructed from an HMR model [63, 47] and b) human pose with contextual appearance as computed by an MViT model [35].

Given a video with number of frames T, we first track every person using PHALP [110], which gives us a unique identity for each person over time. Let a person $i \in [1, 2, 3, ...n]$ at time $t \in [1, 2, 3, ...T]$ be represented by a person-vector $\mathbf{H}_t^i$. Here n is the number of people in a frame. This person-vector is constructed such that, it contains human-centric representation $\mathbf{P}_t^i$ and some contextualized appearance information $\mathbf{Q}_t$.

$$\mathbf{H}_t^i = \{\mathbf{P}_t^i, \mathbf{Q}_t^i\}. \tag{3.1}$$

Since we know the identity of each person from the tracking, we can create an action-tube [46] representation for each person. Let $\mathbf{\Phi}_i$ be the action-tube of person i, then this action-tube contains all the person-vectors over time.

$$\mathbf{\Phi}_i = \{\mathbf{H}_1^i, \mathbf{H}_2^i, \mathbf{H}_3^i, ..., \mathbf{H}_T^i\}. \tag{3.2}$$

Given this representation, we train our model **LART** to predict actions from action-tubes (tracks). In this work we use a vanilla transformer [136] to model the network $\mathcal{F}$, and this allow us to mask attention, if the track is not continuous due to occlusions and failed detections etc. Please see the Appendix for more details on network architecture.

$$\mathcal{F}\big(\mathbf{\Phi}_1, \mathbf{\Phi}_2, ..., \mathbf{\Phi}_i, ..., \mathbf{\Phi}_n; \Theta\big) = \widehat{Y}_i. \tag{3.3}$$

Here, Θ is the model parameters, $\widehat{Y}_i = \{y_1^i, y_2^i, y_3^i, ..., y_T^i\}$ is the predictions for a track, and y_t^i is the predicted action of the track i at time t. The model can use the actions of others for reasoning when predicting the action for the person-of-interest i. Finally, we use binary cross-entropy loss to train our model and measure mean Average Precision (mAP) for evaluation.

Action Recognition with 3D Pose

In this section, we study the effect of human-centric pose representation on action recognition. To do that, we consider a person-vector that only contains the pose representation, $\mathbf{H}_t^i = \{\mathbf{P}_t^i\}$. While, $\mathbf{P}_t^i$ can in general contain any information about the person, in this work train a pose only model **LART-pose** which uses 3D body pose of the person based on the SMPL [83] model. This includes the joint angles of the different body parts, $\theta_t^i \in \mathcal{R}^{23\times3\times3}$ and is considered as an amodal representation, which means we make a prediction about all body parts, even those that are potentially occluded/truncated in the image. Since the global body orientation $\psi_t^i \in \mathcal{R}^{3\times3}$ is represented separately from the body pose, our body representation is invariant to the specific viewpoint of the video. In addition to the 3D pose, we also use the 3D location L_t^i of the person in the camera view (which is also predicted by the PHALP model [110]). This makes it possible to consider the relative location of the different people in 3D. More specifically, each person is represented as,

$$\mathbf{H}_t^i = \mathbf{P}_t^i = \{\theta_t^i, \psi_t^i, L_t^i\}. \tag{3.4}$$

Let us assume that there are n tracklets $\{\mathbf{\Phi}_1, \mathbf{\Phi}_2, \mathbf{\Phi}_3, ..., \mathbf{\Phi}_n\}$ in a given video. To study the action of the tracklet i, we consider that person i as the person-of-interest and having access to other tracklets can be helpful to interpret the person-person interactions for person i. Therefore, to predict the action for all n tracklets we need to make n number of forward passes. If person i is the person-of-interest, then we randomly sample $N-1$ number of other tracklets and pass it to the model $\mathcal{F}(;\Theta)$ along with the $\mathbf{\Phi}_i$.

$$\mathcal{F}(\mathbf{\Phi}_i, \{\mathbf{\Phi}_j | j \in [N]\}; \Theta) = \widehat{Y}_i \tag{3.5}$$

Therefore, the model sees N number of tracklets and predicts the action for the main (person-of-interest) track. To do this, we first tokenize all the person-vectors, by passing them through a linear layer and project it in $f_{proj}(\mathcal{H}_t^i) \in \mathcal{R}^d$ a d dimensional space. Afterward, we add positional embeddings for a) time, b) tracklet-id. For time and tracklet-id we use 2D sine and cosine functions as positional encoding [144], by assigning person i as the zero$^{\text{th}}$ track, and the rest of the tracklets use tracklet-ids $\{1, 2, 3, ..., N-1\}$.

$$\begin{aligned}
\textit{PE}(t, i, 2r) &= \sin(t/10000^{4r/d}) \\
\textit{PE}(t, i, 2r+1) &= \cos(t/10000^{4r/d}) \\
\textit{PE}(t, i, 2s + D/2) &= \sin(i/10000^{4s/d}) \\
\textit{PE}(t, i, 2s + D/2 + 1) &= \cos(i/10000^{4s/d})
\end{aligned}$$

Here, t is the time index, i is the track-id, $r, s \in [0, d/2)$ specifies the dimensions and D is the dimensions of the token.

After adding the position encodings for time and identity, each person token is passed to the transformer network. The $(t + i \times N)^{th}$ token is given by,

$$token_{(t+i\times N)} = f_{proj}(\mathcal{H}_t^i) + \textit{PE}(t, i, :) \tag{3.6}$$

Our person of interest formulation would allow us to use other actors in the scene to make better predictions for the main actor. When there are multiple actors involved in the scene, knowing one person's action could help in predicting another's action. Some actions are correlated among the actors in a scene (e.g. *dancing*, *fighting*), while in some cases, people will be performing reciprocal actions (e.g. *speaking* and *listening*). In these cases knowing one person's action would help in predicting the other person's action with more confidence.

Actions from Appearance and 3D Pose

While human pose plays a key role in understanding actions, more complex actions require reasoning about the scene and context. Therefore, in this section, we investigate the benefits of combining pose and contextual appearance features for action recognition and train model **LART** to benefit from 3D poses and appearance over a trajectory. For every track, we run a 2D action recognition model (*i.e*. MaskFeat [146] pretrained MViT [35]) at a frequency f_s and store the feature vectors before the classification layer. For example, consider a track $\mathbf{\Phi}_i$, which has detections $\{D_1^i, D_2^i, D_3^i, ..., D_T^i\}$. We get the predictions form the 2D action recognition models, for the detections at $\{t, t+f_{FPS}/f_s, t+2f_{FPS}/f_s, ...\}$. Here, f_{FPS} is the rate at which frames appear on the screen. Since these action recognition models capture temporal information to some extent, $\mathbf{Q}^i_{t-f_{FPS}/2f_s}$ to $\mathbf{Q}^i_{t+f_{FPS}/2f_s}$ share the same appearance features. Let's assume we have a pre-trained action recognition model $\mathcal{A}$, and it takes a sequence of frames and a detection bounding box at mid-frame, then the feature vectors for $\mathbf{Q}^i_t$ is given by:

$$\mathcal{A}\big(D_t^i, \{I\}_{t-M}^{t+M}\big) = \mathbf{U}_t^i$$

Here, $\{I\}_{t-M}^{t+M}$ is the sequence of image frames, $2M$ is the number of frames seen by the action recognition model, and $\mathbf{U}_t^i$ is the contextual appearance vector. Note that, since the action recognition models look at the whole image frame, this representation implicitly contains information about the scene and objects and movements. However, we argue that human-centric pose representation has orthogonal information compared to feature vectors taken from convolutional or transformer networks. For example, the 3D pose is a geometric representation while $\mathbf{U}_t^i$ is more photometric, the SMPL parameters have more priors about human actions/pose and it is amodal while the appearance representation is learned from raw pixels. Now that we have both pose-centric representation and appearance-centric representation in the person vector $\mathbf{H}_t^i$:

$$\mathbf{H}_t^i = \{\underbrace{\theta_t^i, \psi_t^i, L_t^i}_{\mathbf{P}_t^i}, \underbrace{\mathbf{U}_t^i}_{\mathbf{Q}_t^i}\} \tag{3.7}$$

So, each human is represented by their 3D pose, 3D location, and with their appearance and scene content. We follow the same procedure as discussed in the previous section to add positional encoding and train a transformer network $\mathcal{F}(\Theta)$ with pose+appearance tokens.

Dataset	# clips	# tracks	# bbox
AVA [51]	184k	320k	32.9m
Kinetics [66]	217k	686k	71.4m
Total	400k	1m	104.3m

Table 3.1: **Tracking statistics on AVA [51] and Kinetics-400 [66]:** We report the number tracks returned by PHALP [110] for each datasets (m: million). This results in over 900 hours of tracks, with a mean length of 3.4 seconds (with overlaps).

3.4 Experiments

We evaluate our method on AVA [51] in various settings. AVA [51] poses an action detection problem, where people are localized in a spatio-temporal volume with action labels. It provides annotations at 1Hz, and each actor will have 1 pose action, up to 3 person-object interactions (optional), and up to 3 person-person interaction (optional) labels. For the evaluations, we use AVA v2.2 annotations and follow the standard protocol as in [51]. We measure mean average precision (mAP) on 60 classes with a frame-level IoU of 0.5. In addition to that, we also evaluate our method on AVA-Kinetics [80] dataset, which provides spatio-temporal localized annotations for Kinetics videos.

We use PHALP [110] to track people in the AVA dataset. PHALP falls into the tracking-by-detection paradigm and uses Mask R-CNN [53] for detecting people in the scene. At the training stage, where the bounding box annotations are available only at 1Hz, we use Mask R-CNN detections for the in-between frames and use the ground-truth bounding box for every 30 frames. For validation, we use the bounding boxes used by [97] and do the same strategy to complete the tracking. We ran, PHALP on Kinetics-400 [66] and AVA [51]. Both datasets contain over 1 million tracks with an average length of 3.4s and over 100 million detections. In total, we use about 900 hours length of tracks, which is about 40x more than previous works [64]. See Table 3.1 for more details.

Tracking allows us to train actions densely. Since, we have tokens for each actor at every frame, we can supervise every token by assuming the human action remains the same in a *1 sec* window [51]. First, we pre-train our model on Kinetics-400 dataset [66] and AVA [51] dataset. We run MViT [35] (pretrained on MaskFeat [145]) at 1Hz on every track in Kinetics-400 to generate pseudo ground-truth annotations. Every 30 frames will share the same annotations and we train our model end-to-end with binary cross-entropy loss. Then we fine-tune the pretrained model, with *tracks* generated by us, on AVA ground-truth action labels. At inference, we take a track, and randomly sample $N-1$ of other tracks from the same video and pass it through the model. We take an average pooling on the prediction head over a sequence of 12 frames, and evaluate at the center-frame. For more details on model architecture, hyper-parameters, and training procedure/training-time please see Appendix.

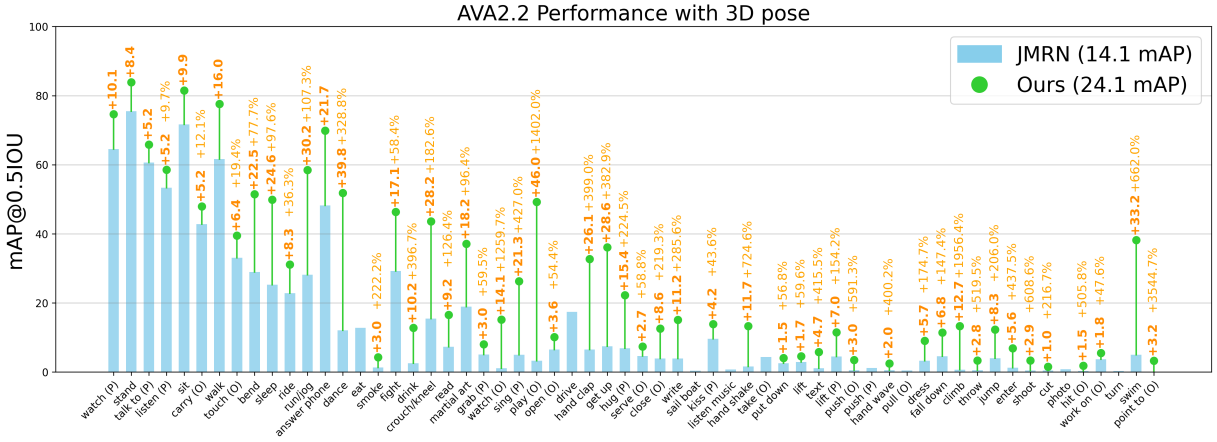


Figure 3.2: **Class-wise performance on AVA:** We show the performance of JMRN [119] and LART-pose on 60 AVA classes (average precision and relative gain). For pose based classes such as *standing*, *sitting*, and *walking* our 3D pose model can achieve above 60 mAP average precision performance by only looking at the 3D poses over time. By modeling multiple trajectories as input our model can understand the interactions among people. For example, activities such as *dancing* (+30.1%), *martial art* (+19.8%) and *hugging* (+62.1%) have large relative gains over state-of-the-art pose only model. We only plot the gains if it is above or below 1 mAP.

Action Recognition with 3D Pose

In this section, we discuss the performance of our method on AVA action recognition, when using 3D pose cues, corresponding to Section 3.3. We train our 3D pose model **LART-pose**, on Kinetics-400 and AVA datasets. For Kinetics-400 tracks, we use MaskFeat [146] pseudo-ground truth labels and for AVA tracks, we train with ground-truth labels. We train a single person model and a multi-person model to study the interactions of a person over time, and person-person interactions. Our method achieves 24.1 mAP on multi-person (N=5) setting (See Table 3.2). While this is well below the state-of-the-art performance, this is a first time a 3D model achieves more than 15.6 mAP on AVA dataset. Note that the first reported performance on AVA was 15.6 mAP [51], and our 3D pose model is already above this baseline.

We evaluate the performance of our method on three AVA sub-categories (Object Manipulation (*OM*), Person Interactions (*PI*), and Person Movement(*PM*)). For the person-movement task, which includes actions such as *running*, *standing*, and *sitting* etc., the 3D pose model achieves 48.7 mAP. In contrast, MaskFeat performance in this sub-category is 58.6 mAP. This shows that the 3D pose model can perform about 80% good as a strong state-of-the-art model. On the person-person interaction category, our multi-person model achieves a gain of +2.1 mAP compared to the single-person model, showing that the multi-person model was able to capture the person-person interactions. As shown in the Fig 3.2, for person-person interactions classes such as *dancing*, *fighting*, *lifting a person* and *handshaking* etc., the multi-person model performs much better than the current state-of-the-art pose-only models. For example, in *dancing* multi-person model gains

Model	Pose	OM	PI	PM	mAP
PoTion [24]	2D	-	-	-	13.1
JMRN [119]	2D	7.1	17.2	27.6	14.1
LART-pose	3D (n=1)	12.0	22.0	46.6	22.9
LART-pose	3D (n=5)	13.3	25.9	48.7	24.1

Table 3.2: **AVA Action Recognition with 3D pose:** We evaluate human-centric representation on AVA dataset [51]. Here *OM* : Object Manipulation, *PI* : Person Interactions, and *PM* : Person Movement. LART-posecan achieve about 80% performance of MViT models on person movement tasks without looking at scene information.

+39.8 mAP, and in *hugging* the relative gain is over +200%. In addition to that, the multi person model has the largest gain compared to the single person model in the person interactions catagory.

On the other hand, object manipulation has the lowest score among these three tasks. Since we do not model objects explicitly, the model has no information about which object is being manipulated and how it is being associated with the person. However, since some tasks have a unique pose when interacting with objects such as *answering a phone* or *carrying an object*, knowing the pose would help in identifying the action, which results in 13.3 mAP.

Actions from Appearance and 3D Pose

While the 3D pose model can capture about 50% performance compared to the state-of-the-art methods, it does not reason about the scene context. To model this, we concatenate the human-centric 3D representation with feature vectors from MaskFeat [146] as discussed in Section 3.3. MaskFeat has a MViT2 [81] as the backbone and it learns a strong representation about the scene and contextualized appearance. First, we pretrain this model on Kinetics-400 [66] and AVA [51] datasets, using the pseudo ground truth labels. Then, we fine-tune this model on AVA tracks using the ground-truth action annotation.

In Table 3.3 we compare our method with other state-of-the-art methods. Overall our method has a gain of **+2.8 mAP** compared to Video MAE [40, 130]. In addition to that if we train with extra annotations from AVA-Kinetics our method achieves **42.3 mAP**. Figure 3.3 show the class-wise performance of our method compared to MaskFeat [145]. Our method overall improves the performance of 56 classes in 60 classes. For some classes (e.g. *fighting*, *hugging*, *climbing*) our method improves the performance by more than +5 mAP. In Table 3.4 we evaluate our method on AVA-Kinetics [80] dataset. Compared to the previous state-of-the-art methods our method has a gain of +1.5 mAP.

In Figure 3.4, we show qualitative results from MViT [35] and our method. As shown in the figure, having explicit access to the tracks of everyone in the scene allow us to make more confident predictions for actions like *hugging* and *fighting*, where it is easy to interpret close interactions. In addition to that, some actions like *riding a horse* and *climbing* can benefit from having access to

explicit 3D poses over time. Finally, the amodal nature of 3D meshes also allows us to make better predictions during occlusions.

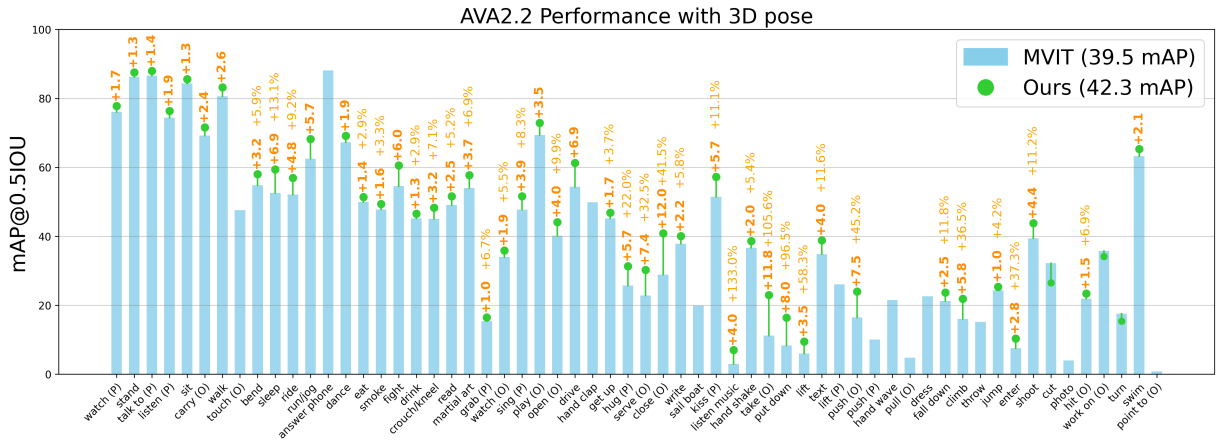


Figure 3.3: **Comparison with State-of-the-art methods:** We show class-level performance (average precision and relative gain) of MViT [35] (pretrained on MaskFeat [146]) and ours. Our methods achieve better performance compared to MViT on over 50 classes out of 60 classes. Especially, for actions like *running*, *fighting*, *hugging*, and *sleeping* etc., our method achieves over **+5 mAP**. This shows the benefit of having access to explicit tracks and 3D poses for action recognition. We only plot the gains if it is above or below 1 mAP.

Ablation Experiments

Effect of tracking: All the current works on action recognition do not associate people over time, explicitly. They only use the mid-frame bounding box to predict the action. For example, when a person is *running* across the scene from left to right, a feature volume cropped at the mid-frame bounding box is unlikely to contain all the information about the person. However, if we can track this person we could simply know their exact position over time and that would give more localized information to the model to predict the action.

To this end, first, we evaluate MaskFeat [145] with the same detection bounding boxes [97] used in our evaluations, and it results in 40.2 mAP. With this being the baseline for our system, we train a model which only uses MaskFeat features as input, but over time. This way we can measure the effect of tracking in action recognition. Unsurprisingly, as shown in Table 3.5 when training MaskFeat with tracking, the model performs +1.2 mAP better than the baseline. This clearly shows that the use of tracking is helpful in action recognition. Specifically, having access to the tracks help to localize a person over time, which in return provides a second order signal of how joint angles changes over time. In addition, knowing the identity of each person also gives a discriminative signal between people, which is helpful for learning interactions between people.

Model	Pretrain	mAP
SlowFast R101, 8×8 [41]	K400	23.8
MViTv1-B, 64×3 [35]		27.3
SlowFast 16×8 +NL [41]	K600	27.5
X3D-XL [38]		27.4
MViTv1-B-24, 32×3 [35]		28.7
Object Transformer [150]		31.0
ACAR R101, 8×8 +NL [97]		31.4
ACAR R101, 8×8 +NL [97]	K700	33.3
MViT-L↑312, 40×3 [81],	IN-21K+K400	31.6
MaskFeat [146]	K400	37.5
MaskFeat [146]	K600	38.8
Video MAE [40, 130]	K600	39.3
Video MAE [40, 130]	K400	39.5
LART	K400	42.3 (+2.8)

Table 3.3: **Comparison with state-of-the-art methods on AVA 2.2:**. Our model uses features from MaskFeat [146] with full crop inference. Compared to Video MAE [40, 130] our method achieves a gain of **+2.8 mAP**.

Model	mAP
SlowFast [41]	32.98
ACAR [97]	36.36
RM [42]	37.34
LART	**38.91**

Table 3.4: **Performance on AVA-Kinetics Dataset.** We evaluate the performance of our model on AVA-Kinetics [80] using a single model (no ensembles) and compare the performance with previous state-of-the-art single models.

Effect of Pose: The second contribution from our work is to use 3D pose information for action recognition. As discussed in Section 3.4 by only using 3D pose, we can achieve 24.1 mAP on AVA dataset. While it is hard to measure the exact contribution of 3D pose and 2D features, we compare our method with a model trained with only MaskFeat and tracking, where the only difference is the use of 3D pose. As shown in Table 3.5, the addition of 3D pose gives a gain of +0.8 mAP. While this is a relatively small gain compared to the use of tracking, we believe with more robust and accurate 3D pose systems, this can be improved.

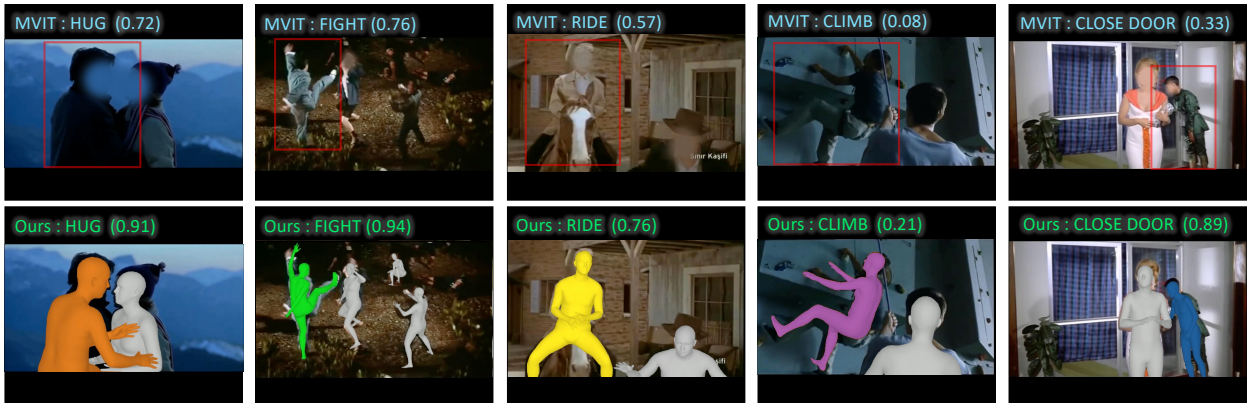


Figure 3.4: **Qualitative Results:** We show the predictions from MViT [35] and our model on validation samples from AVA v2.2. The person with the colored mesh indicates the person-of-interest for which we recognise the action and the one with the gray mesh indicates the supporting actors. The first two columns demonstrate the benefits of having access to the action-tubes of other people for action prediction. In the first column, the orange person is very close to the other person with hugging posture, which makes it easy to predict *hugging* with higher probability. Similarly, in the second column, the explicit interaction between the multiple people, and knowing others also fighting increases the confidence for the *fighting* action for the green person over the 2D recognition model. The third and the fourth columns show the benefit of explicitly modeling the 3D pose over time (using tracks) for action recognition. Where the yellow person is in riding pose and purple person is looking upwards and legs on a vertical plane. The last column indicates the benefit of representing people with an amodal representation. Here the hand of the blue person is occluded, so the 2D recognition model does not see the action as a whole. However, SMPL meshes are amodal, therefore the hand is still present, which boosts the probability of predicting the action label for *closing the door*.

Implementation details

In both the pose model and pose+appearance model, we use the same vanilla transformer architecture [136] with 16 layers and 16 heads. For both models the embedding dimension is 512. We train with 0.4 mask ratio and at test time use the same mask token to in-fill the missing detections. The output token from the transformer is passed to a linear layer to predict the AVA action labels. We pre-train our model on kinetics for 30 epochs with MViT [35] predictions as pseudo-supervision and then fine-tune on AVA with AVA ground truth labels for few epochs. We train our models with AdamW [84] with base learning rate of 0.001 and betas $= (0.9, 0.95)$. We use cosine annealing scheduling with a linear warm-up. For additional details please see the Appendix.

Model	OM	PI	PM	mAP
MViT	32.2	41.1	58.6	40.2
MViT + Tracking	33.4	43.0	59.3	41.4 (+1.2)
MViT + Tracking + Pose	34.4	43.9	59.9	42.3 (+0.9)

Table 3.5: **Ablation on the main components:** We ablate the contribution of tracking and 3D poses using the same detections. First, we only use MViT features over the tracks to evaluate the contribution from tracking. Then we add 3D pose features to study the contribution from 3D pose for action recognition.

3.5 Conclusion

In this chapter, we investigated the benefits of 3D tracking and pose for the task of human action recognition. By leveraging a state-of-the-art method for person tracking, PHALP [110], we trained a transformer model that takes as input tokens the state of the person at every time instance. We investigated two design choices for the content of the token. First, when using information about the 3D pose of the person, we outperform previous baselines that rely on pose information for action recognition by 8.2 mAP on the AVA v2.2 dataset. Then, we also proposed fusing the pose information with contextualized appearance information coming from a typical action recognition backbone [35] applied over the tracklet trajectory. With this model, we improved upon the previous state-of-the-art on AVA v2.2 by 2.8 mAP. There are many avenues for future work and further improvements for action recognition. For example, one could achieve better performance for more fine-grained tasks by more expressive 3D reconstruction of the human body (*e.g.*, using the SMPL-X model [100] to capture also the hands), and by explicit modeling of the objects in the scene (potentially by extending the "tubes" idea to objects).

Acknowledgements: This work was supported by the FAIR-BAIR program as well as ONR MURI (N00014-21-1-2801). We thank Shubham Goel, for helpful discussions.

3.6 Additional Results

We include additional experiments and implementation details about our approach (Sections 2 & 3), we provide more details about the experiments of our chapter (Sections 4) and we include the training configurations for Kinetics-400 and AVA for reproducibility.

Modeling Multiple People in the Scene

As discussed in Section 3.3 our model take any number of people (tracks) given enough memory. Even though we do simple random sampling to find supporting actors in the scene, knowing this additional context of where other people are located and what they are doing could be a strong signal to predict what the person of interest is doing. We train multiple models, by varying the maximum context for people ($n \in [1, 2, 3, 4, 5]$). When $n = 1$, the LARTonly sees the person-of-interest and the information about the scene and other people are fed through the contextualized appearance vector. However, with larger n, other people's poses, locations and appearance are explicitly given to the model. As shown in the Figure 3.5 as we increase the LART-pose model's people-context, the performance on AVA dataset increases monotonically, and starts saturating at $n > 4$.

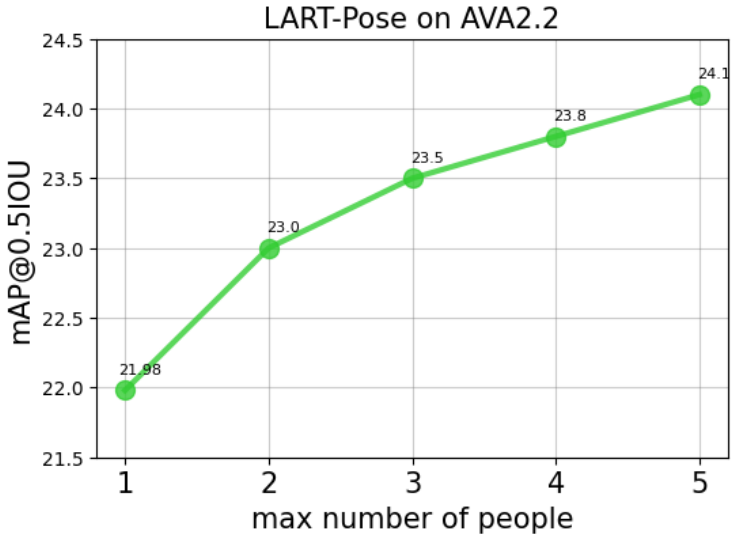


Figure 3.5: **LART-pose performance with number of people in the scene:** We show the performance of LART-pose on various n (maximum number of people the model sees in every frame). This plot shows that as we increase the number of people the model can see while reasoning about the person of interest, the performance increase monotonically. While our multi-person model is very simple; we just input additional tokens for each other people in the scene, the model is able to understand these interactions from the large scale training data.

This observation highlights the benefits of modeling multiple people in the scene and learning their interactions jointly. However, it is only possible to achieve these benefits if we can track

every person in the scene. We observed saturation at $n = 5$ with simple random sampling, but this observation may be heavily biased for the AVA dataset. Most movies have few characters, and the biases in the way movies are captured (e.g., close-ups for kissing and hugging scenes) will have an impact on these results. It's important to note that all of these models have the same number of parameters and were trained for the same amount of time (30 epochs on the same dataset).

In summary, our findings suggest that modeling multi-person interactions can significantly improve the performance of action recognition models, particularly for actions that involve close person-person interactions or group activities. However, the saturation point may vary depending on the dataset and biases in the way scenes are captured.

Single person vs Multi person Results

In the previous section, we discussed the benefits of modeling multi-person interactions. In this section, we will study how much performance gain can be achieved from a single person model to a multi-person model. We compare the performance of LART-pose (n=1) and LART-pose (n=5) in Figure 3.6. The multi-person model has an overall gain of **2.0 mAP** over the single person model.

Upon closer examination of the performance of each class, it becomes apparent that classes involving close person-person interactions benefited greatly from multi-person training. For example, actions such as *hugging*, *kissing*, *handshaking*, *lifting a person*, and *listening to a person* have improved by over **10%** relative to the single person model. These interactions occur at very close proximity, and having explicit knowledge of the other person's 3D location, pose, and action would aid in identifying the actions of the main actor.

In addition to the close interaction actions, there are group actions that would also benefit from using the context of other people's actions to reason about the action of the person of interest. For example, *dancing* and *swimming* are typically group activities, and knowing what others are doing is a good signal to infer the action.

From this multi-person model, over 50 classes have gained over the single person model, and over 30 classes have gained over **1 mAP**. However, as mentioned in the previous section regarding biases in datasets, these results may vary slightly for different types of datasets, such as sports datasets. For example, in a sports scene, it may be necessary to look at more than 5 people to recognize the action of the player, and the way sports scenes are shot is significantly different from movies.

Overall, our findings demonstrate that modeling multi-person interactions can significantly improve the performance of action recognition models, particularly for actions that involve close person-person interactions or group activities.

3.7 Implementation details

Our complete system for action recognition by tracking integrates multiple sub-systems to combine the recent advancements in 2D Detection, Recognition, 3D Reconstruction, as well as Tracking. We can break the overall pipeline into two parts: **a)** frames-to-entities and **b)** entities-to-action.

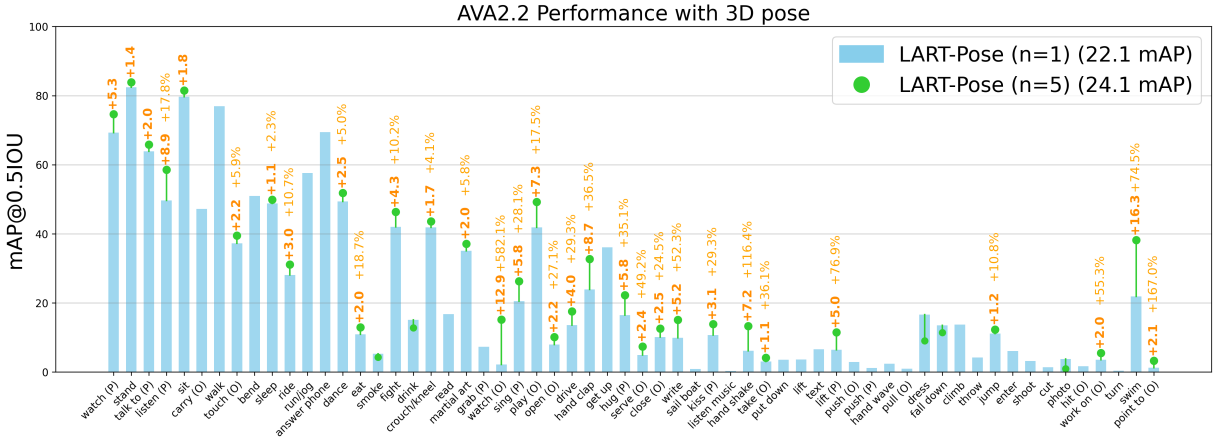


Figure 3.6: **Class-wise performance on AVA:** We compare LART-poseon single person $(n = 1)$ and multi-person $(n = 5)$ setting. Our multi-person model outperforms single person model on over 50 classes and on some person-person interaction classes multi-person model has a relative gain of about **10%**.

The first part is to lift entities from frames (here, we consider entities=people). For this, we use a state-of-the-art tracking algorithm, PHALP [110]. The first step of PHALP is to detect people in each frame using Mask R-CNN [53]. We used Detectron2's [151] new baseline models trained with Simple Copy-Paste Data Augmentation [43] with a RegNet-4gf [104] backbone for the detection task. After detecting people in each frame, PHALP uses HMR [63, 111, 47] to reconstruct each person in 3D. Then, the future location, pose, and appearance of each person are predicted for solving association. PHALP uses Hungarian matching to solve the associations between 3D detections and 3D predictions. Finally, a set of tracks will be returned from the tracking which gives us access to entities (people) over time.

For the second part of this chapter, we collect tracks and use them to train a transformer model for action recognition. More details on the network architecture and training and inference protocols will be discussed in the following sections. This part of the chapter is a crucial component of our approach to using transformer models for action recognition from 3D tracks.

PHALP tracklets

In this work, every person in both Kinetics-400 [66] and AVA [51] is tracked. For this, we used the recently proposed 3D tracking algorithm PHALP [110]. PHALP allows us to track people in the wild very robustly and gives their 3D representations. However, the ground-truth action annotations for AVA are given as bounding boxes at 1 Hz frequency. On a side note, we do not use the ground truth tracking annotations in AVA dataset, which is also only available at 1 Hz. First, we use the PHALP detection model (*e.g.*, Mask R-CNN) to detect humans in the video, whenever a frame does not have ground-truth annotations. If the frame indeed has an annotation, we take the ground-truth bounding boxes as granted and bypass Mask R-CNN detections. Since AVA only has bounding box annotations, and PHALP [110] requires bounding boxes and masks, we use Detectron2 [151] to extract masks from bounding boxes with Mask R-CNN. For the validation set, we used the detections from ACAR [97], which are also only available every 30 frames. Therefore, we used a similar strategy to get tracks from bounding boxes available at 1 Hz. For Kinetics, we run PHALP tracking for the whole sequence, which is typically 10s clips. However, since AVA is much longer than Kinetics (15 min), we run the tracker for 4-second windows, centered around the evaluation frame.

Architecture details

In all of our experiments, we use a vanilla transformer [136] architecture with 16 layers and width of 512. Each layer has 16 self-attention heads followed by layer-norm [7], and a 2-layer MLP followed by layer-norm. We train all the models with a maximum sequence length of 128 frames per person. In other words, every tracklet is trimmed to have a sequence length of 128 frames. The only data augmentation we use is choosing the starting point of the sequence for random trimming. The transformer blocks are followed by a linear layer that predicts AVA action classes. We train all our models with binary cross-entropy loss.

At training time, we use two types of attention masking. First, since the tracklets are not always continuous due to occlusion and missing detections, we mask the corresponding self-attention of these tokens completely. The loss is not applied to these tokens and this part of the tracklet has no effect on training. The second type of masking is done to simulate these kinds of missing detections at test time. We randomly choose a small number of tokens (based on mask ratio), and replace the person-vector with a learnable mask-token. At the self-attention layer, attention is masked such that these masked-tokens will attend other tokens but other tokens will not attend the masked-tokens. Unlike, the first type of masking, we apply loss on these masked token predictions, since if there is a detection available, then there will be a pseudo-ground truth or ground truth label available for training.

At inference time, we do not do any attention-masking. However, there will be some tracklets with discontinuous detections. At these locations, we use the learned masked-token to infill the predictions for the tracklets. Since we are predicting action labels densely for each frame, we take an average pooling of 12 tokens centered around the annotated detection to minimize the gap between human annotations and model predictions.

Action with 3D Pose

In this subsection, we discuss the network architecture used for recognizing action only with 3D pose information over time. The 3D pose has 226 parameters: 207 ($23 \times 3 \times 3$) parameters for joint angles, 9 for the global orientation of the person, and 10 for the body shape. In addition to this, the 3D translation of the person in the camera frame is represented by 3 parameters. Overall, in this system, a person-vector has a dimension of 229. This vector is encoded by an MLP with two hidden layers to project this to a 256-dimensional vector. The projected person-vector is then passed to the transformer. We also use the three types of positional encodings for time, track, and space as discussed in the main manuscript (Section 3.1).

Action with 3D Pose and Appearance

To encode a strong contextualized appearance feature, we used MViT [35] pretrained with MaskFeat [145]. The MViT model for AVA takes a sequence of frames and a mid-frame bounding box to predict the action label of the person of interest (this is a classical example of the Eulerian way of predicting action). In this chapter, we use an MViT-L 40×3 model that takes a 4-second clip and samples 40 frames with a temporal stride of 3 frames as the input and a bounding box of a person at the mid-frame. This gives a 1152-dimensional feature vector before the linear layer in the MViT classifier. We use this 1152-dimensional feature vector as our contextualized appearance feature and encode it into a 256 dimensional vector by an MLP with two hidden layers. Now, we have a pose vector (256 dim, from the previous section) and an appearance vector (256 dim). We concatenate these two vectors to build our person-vector for 3D pose with appearance, and the final 512-dimensional vector is passed to the transformer.

Training recipe

As discussed in Section 3 of the main manuscript, we first pretrain our method on Kinetics-400 dataset, using the tracklets obtained from PHALP [110]. Each of these tracklets contains a detection at every frame unless the person is occluded or is not detected due to failure of the detection system. We provide these detection bounding boxes as input to the MViT [35] model and generate pseudo ground truth action labels for the tracklets. Once the labels are generated, we train our model end-to-end, with tracklets as inputs and the action labels as outputs. We use the training configurations in Table 3.6 for pretraining the model on Kinetics-400 tracklets. Once the model is pretrained on Kinetics tracklets, we fine-tune the model on AVA tracklets (generated by PHALP) with ground truth action labels. Finally, during the fine-tuning stage, we apply layer-wise decay [26] and drop path [57].

Configs	Kinetics-400	AVA
optimizer	AdamW [84]	
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.95$	
weight decay	0.05	
learning rate schedule	cosine decay [85]	
warmup epochs	5	
drop out	0.1	
base learning rate	1e-3	1e-3
layer-wise decay [26]	-	0.9
batch size	64	64
training epochs	30	30
drop path [57]	-	0.1
mask ratio	0.4	0.0

Table 3.6: **Training Configurations:** We report the training configurations used from training our models on Kinetics-400 and AVA datasets.

Chapter 4

An Empirical Study of Autoregressive Pre-training from Videos

We empirically study autoregressive pre-training from videos. To perform our study, we construct a series of autoregressive video models, called *Toto*. We treat videos as sequences of visual tokens and train transformer models to autoregressively predict future tokens. Our models are pre-trained on a diverse dataset of videos and images comprising over 1 trillion visual tokens. We explore different architectural, training, and inference design choices. We evaluate the learned visual representations on a range of downstream tasks including image recognition, video classification, object tracking, and robotics. Our results demonstrate that, despite minimal inductive biases, autoregressive pre-training leads to competitive performance across all benchmarks. Finally, we find that scaling our video models results in similar scaling curves to those seen in language models, albeit with a different rate.

4.1 Introduction

In a paper published in 1951, Shannon, having just published the foundational papers of information theory, proposed a "guessing game" of *next word prediction* to estimate the entropy of English [121]. Nearly 70 years later, training a high-capacity transformer network [136] on this task, provided the generative pre-training backbone for Large Language Models [102, 30, 103, 15].

Less well known is the fact that in 1954, Fred Attneave [6] proposed an analog of Shannon's task for images. To quote "We may divide the picture into arbitrarily small elements which we "transmit" to a subject (S) in a cumulative sequence, having them guess at the color of each successive element until they are correct. This method of analysis resembles the scanning process used in television and facsimile systems and accomplishes the like purpose of transforming two spatial dimensions into a single sequence in time".

While Attneave was concerned with images, in the context of 2024, we have to note that the "Big Visual Data" is in videos. While there are concerns that most of the text available on the Internet has already been used by the language models, in video we just started on the journey of

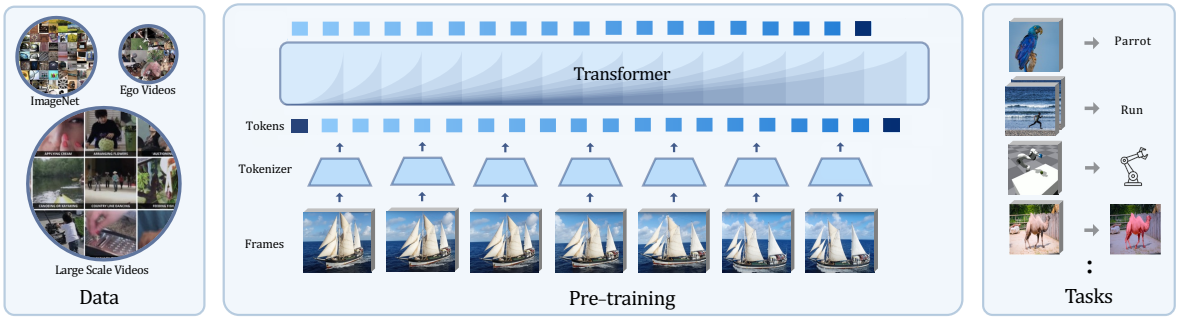


Figure 4.1: **Overall Framework.** Starting with images and video frames from a collection of datasets, we tokenize each frame/image into discrete visual tokens independently. We pre-train the transformer by predicting the next visual tokens, with a context length of 4K tokens of images or video frames. Once trained, we take the intermediate representations and evaluate them on various tasks.

Big Data exploitation. Despite the successes of autoregressive language and image models, their effectiveness for video modeling remains underexplored.

In this chapter, we empirically study autoregressive pre-training from videos. To perform our empirical study, we construct a family of autoregressive video models which we call *Toto*. We treat videos as sequences of visual tokens and train a causal transformer models on next-token prediction task. We use causal transformer model with LLaMa [131] architecture. We use dVAE [112] to tokenize frames into discrete tokens. Treating videos as sequences of tokens enables us to jointly train on videos and images using a unified format. We construct a diverse dataset of videos and images comprising over 1 trillion visual tokens. Our models are first pre-trained on this data and then evaluated on downstream tasks. We extract visual representations using attention pooling from relevant layers of the model.

We evaluate our models on various downstream tasks from image and video recognition, video forecasting, semi-supervised tracking, object permanence and robotics tasks in both simulation and real-world. We consider different design choices such as tokenizers including dVAE [112], VQ-GAN [116] and continuous patch-normalized [54] tokens. We also consider different architectures such as LLaMA [131], GPT2 [103] and Mamba [50]. Finally we study the compute optimal scaling behaviors of autoregressive video models.

We find that, for tokenization autoregressive models based on discrete and continuous patch-normalized [54] tokens perform similarly on ImageNet classification task. For efficient pre-training, starting with lower resolution and fine-tuning at higher resolution gives better performance and RoPE [126] helps with adopting to higher resolution. For measuring the representation quality in decoder-only models, due to skewed nature of the receptive field we use attention pooling over average pooling. We find that in decoder-only models, for all tasks and models sizes the middle layer gives the best performance. Finally, we study the scaling behaviors of autoregressive vision models, which scales with more compute but still at a slower rate compared to large language models.

4.2 Related work

Representation Learning for Vision: Over the years self-supervised pre-training has proven to be effective in many areas including language, vision, and robotics. [152] and SimCLR [21] showed that instance discrimination training can learn strong discriminative features. MoCo [55] and DINO [16] showed the effectiveness of strong visual representations on various downstream tasks. Differently, BEiT [8] and MAE [54] used masked autoencoding for learning image representations. ST-MAE [40]and VideoMAE [139] extended this masked modeling approach to videos, by masking a large amount of tokens during pre-training and predict the masked tokens with a light-weight decoder.

Autoregressive Modeling of Vision: Generative autoregressive pre-training learns to directly model the data distribution. In language models, generative pre-training has become the standard for training large models. For autoregressive pre-training in vision, rCNN [113], PixelCNN [133] and PixelRNN [134] proposed generating pixels one by one using convolution and bidirectional LSTMs. With the introduction of the transformers [136], ImageTransformers [98] showed generating pixels with causal local attention performs better than previous CNN and RNN-based methods. While all of these methods focused on the generation quality of the pixels, iGPT [20] showed that generative pre-training is also a good way to learn strong visual representations for recognition tasks. [56] showed scaling behaviors of autoregressive image and video models. AIM [94] on the other hand uses patch embedding rather than any pre-trained models for tokenization, however, it trains on Data Filtering Networks [36] with clip filtered data. Compared to these works, we do not use any supervision during our pre-training and utilizes image and videos jointly. VisionMamba [168] also showed how to utilize sequence models with bidirectional state-space modeling for supervised vision tasks. [148] showed autoregressive video generation for promotable video generations.

Evaluation of Vision Representations: Most video pre-training models are evaluated on semantic tasks like ImageNet [29] and Kinetics [66]. Additionally to the standard evaluation, we evaluate our models on semi-supervised tracking task on DAVIS [101], action forecasting on Ego4D [48], object permanence on CATER [44] and on robot manipulation tasks in simulation [154] and in the real world [106].

4.3 Approach

We train a casual transformer model to predict the next patch tokens in images and videos. This is akin to the next token prediction in large language models. From the vast collection of images and videos, every patch is tokenized into a discrete token, and the transformer is trained to predict the next token, using raster scan ordering. We pre-train our models on over one trillion tokens. Finally, we evaluate the learned representations of these models on various downstream tasks including image classification, action classification, action anticipation, video tracking, object permanence, and robotic manipulation tasks. We also study the scaling behaviors of our models for compute optimal training.

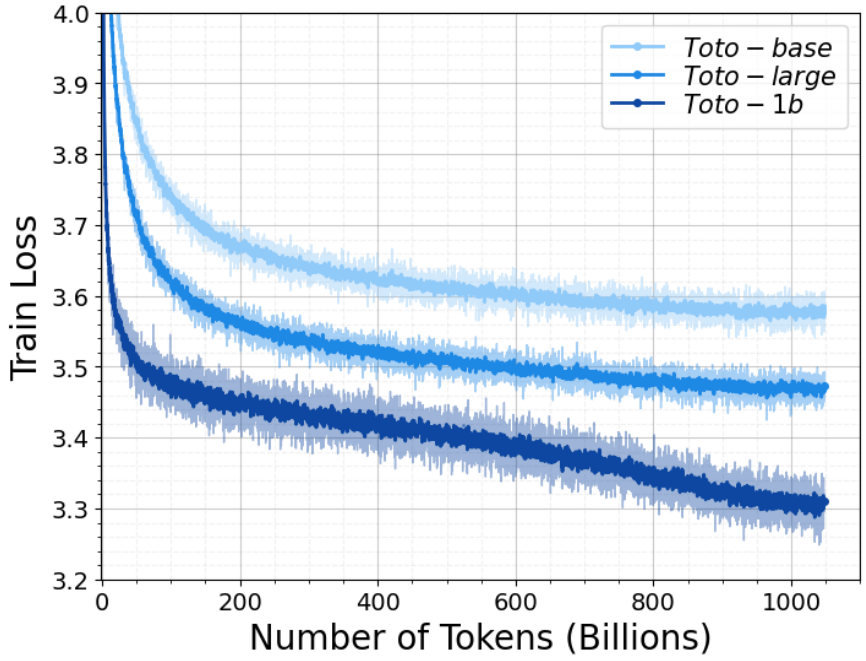


Figure 4.2: **Training Loss Curves:** We show the training loss curves for base, large, and 1b models trained with tokens from dVAE [112] with a vocabulary size of 8k and context length of 4k tokens (equivalent to 16 images or video frames).

Pre-training

Given a large collection of images and videos, we tokenize all of them into a 1D sequence using raster scan ordering. This produces a dataset of tokens, $\{x_1^j, x_2^j, x_3^j, ..., x_n^j\}$ where j is the sample either from a video or an image and n is the number of tokens in an image or a video. We model the density $p(x)$ as :

$$p(x^j) = \prod_{i=1}^{n} p(x_i^j | x_{i-1}^j, x_{i-2}^j, ..., x_1^j, \theta) \tag{4.1}$$

Here, θ is the model parameters, which can be optimized by minimizing the negative log-likelihood loss:

$$\mathcal{L}_{\text{pre-train}} = \mathop{\mathbb{E}}_{x^j \sim X} - \log p(x^j). \tag{4.2}$$

Using this loss, we pre-train our models at different sizes on over one visual trillion tokens. These tokens are generated from images and video. Figure 4.2 shows the training loss of 3 differently sized models with 120M, 280m and 1.1b parameters.

Architecture

Our model is a transformer [136] with causal attention. We apply recent advancements in language modeling such as pre-norm using RMSNorm [161], SwiGLU activation [122], and RoPE positional embeddings [126], following LLaMa [131].

For a model with L layers, we define H^l to be the intermediate representations after layer $l, 0 \leq l \leq L$. The intermediate representations after layer $l+1$, H^{l+1}, defined to be:

$$\widehat{H}^{l+1} = H^l + \texttt{MHSA}(\texttt{RMS-norm}(H^l)) \tag{4.3}$$

$$H^{l+1} = \widehat{H}^{l+1} + \texttt{MLP}(\texttt{RMS-norm}(\widehat{H}^{l+1})), \tag{4.4}$$

Where `MHSA` is a multi-head self attention layer, `MLP` is a multi-layer perceptron with SwiGLU activations.

We train our models for the next token prediction task at different scales (base, large and 1b models). For more architecture details see Table 4.1. We train all these models with a batch size of 1M tokens. We use AdamW [84] with a maximum learning rate of $3e{-}4$, and $\beta_1 = 0.9, \beta_2 = 0.95$. We decay the learning rate with a cosine schedule, after 2000 warm-up steps [131].

Model	Params	Dimension	Heads	Layers
base	120m	768	12	12
large	280m	1024	16	16
1b	1.1b	2048	16	22

Table 4.1: **Model Architecture**: We pre-train models at different scales, only on visual tokens from images and videos.

Dataset

To train our model, we compile a large dataset from a number of different sources. Table 4.2 shows the total number of images and videos used for training data, the total number of tokens, as well as the number of hours of videos in each dataset. Together these datasets contain over 100,000 hours of video data and about 2.5 trillion visual tokens. During training, each mini-batch is sampled at different ratios of datasets. Each batch approximately contains 20% of ImageNet images, 10% of Ego4D videos, 10% of Kinetics videos, and 60% of HowTo100m videos. Our full training utilized about 1 trillion tokens.

Tokenization

We use dVAE tokenizer with a vocabulary of 8k tokens, from Dall-E [112] as our tokenizer. Using an image-based tokenizer allows training on both images and videos and testing on respective downstream tasks. While VQGAN [34] tokenizers provide sharper images, these models were

Datasets	Instances	Tokens	Hours
ImageNet	13.9M	3.6B	-
Kinetics-600	0.53M	41.3B	1496
Ego4D	52.1K	103B	3750
HowTo100m	1.172M	2560B	92627

Table 4.2: **Pre-training Dataset**: We use both image datasets (Imagenet [117]) and video datasets (Kinetics600 [19], Ego4D [48], HowTo100m [91]) with different mixing ratios during the pre-training of our models. The whole training data contains about 100,000 hours of videos.

trained with perceptual loss [78, 62], thus indirectly ingesting ImageNet label information via VGG-net [124].

All raw pixel frames or images are tokenized into 256 discrete tokens. We take a video and resize it such that its shortest size is R pixels, and then take a random crop of $R \times R \times T$, and sample every 4 frames where T is the number of frames. We use dVAE [112] with the vocabulary of 8k entries to tokenize every frame independently. For dVAE we set $R = 128$, to get 16×16 discrete tokens. Once every frame is mapped into a set of discrete tokens we have $T \times 256$ tokens per each video. We pre-train all the models with $T = 16$, thus all the models are per-trained for a context length of 4096 tokens.

When training with images and videos, 16 video frames are sampled to create 4k tokens. For images, we randomly sample 16 images and create a sequence of 16 image frames to generate 4k tokens. Finally, we add start and end tokens for each sequence, for videos we use `[1]` as the start token, and for images we use `[3]` as the start token, and all sequences have an end token of `[2]`.

Downstream Transfer

The idea of large pre-trained models is that they were trained at a large compute scale, and then these models can be easily used for various downstream tasks without requiring task-specific design or lots of computing for transfer. The learned representations are general enough to transfer to various tasks. We evaluate our models on the intermediate features with linear and attention probing [79].

For linear probing the model, we apply global average pooling [82] over the tokens from different layers to get the intermediate representation. We train a linear layer on top of this representation on the downstream task. MAE [54] or DINO [16] have a uniform structure when it comes to which token attends to which tokens, however in autoregressive sequence modeling later tokens attent to more tokens than the tokens at the beginning. Due to this skewed nature, equally weighting all the tokens affects the downstream performance. Attention pooling is an alternative to average pooling that allows to dynamically weight the tokens, ideally giving more weight to tokens that see more tokens. This requires learning W_k and W_v matrices and a query token q. The query token cross-attends to the intermediate tokens and combines them into a single vector. While this function is not linear anymore, it has been shown to learn better representations in recent works [94].

4.4 Experiments

We evaluate our pre-trained models on various downstream tasks such as ImageNet classification, Kinetics action recognition, Ego4D action anticipation, Semi-Supervised tracking, and Robotic manipulation tasks. First, we discuss various design choices for pre-training and evaluation strategies for our method. All the models for studying the design choices are `large` models trained for 400 epochs on the ImageNet-1k dataset.

Design Choices

Tokenizer: The are various options available for tokenizing an image or a video. We could use discrete tokenizers such as dVAE, and VQGAN, or simple patch-based continuous tokenization. To study the behavior of various tokenizers we pre-train a *Toto*-large model on ImageNet for 400 epochs. Using linear probing at an optimal intermediate layer, we evaluate the accuracy of the models on ImageNet classification task.

Table 4.3 shows linear probing accuracy when trained with various tokenizers. VQGAN [34] and dVAE [112] perform similarly with the same resolutions. However, VQGAN is contaminated with ImageNet label information via perceptual loss. In addition to that, as shown in Figure 4.3, dVAE tokens have full coverage compared to VQGAN tokens on their 1-gram distributions. Please see the supplementary material for more details. Regressing normalized-patch targets from patch embeddings performs slightly worse than classifying discrete tokens as targets. Additionally, discrete tokens as targets and patch embeddings as inputs perform poorly compared to other methods at the given input-output resolutions. Overall, Table 4.3 shows that various ways of tokenization have *little effect* on ImageNet linear probing accuracy.

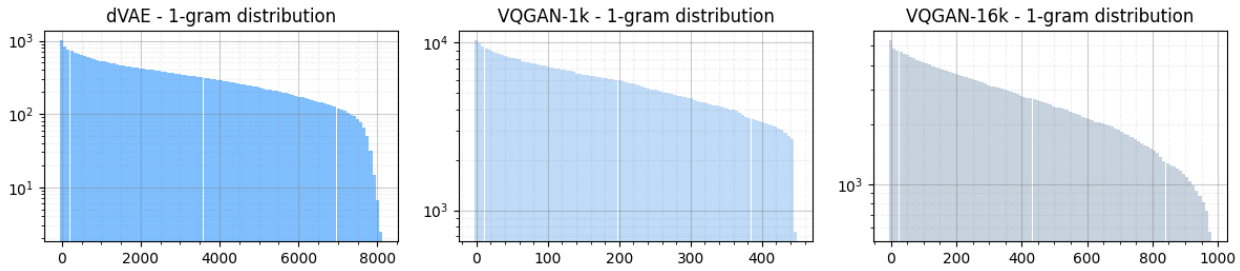


Figure 4.3: **1-gram Distribution of Various Tokens:** This Figure shows the distribution of 1-gram tokens of various tokenizers (dVAE [112], VQGAN-1k, VQGAN-16k [34]) on Imagenet validation set. Note that, dVAE has almost full convergence of the tokens while VQGAN has less than 50% coverage of the tokens.

How to Probe: As discussed in Section 4.3 we probe the pre-trained models at the same layer with attention pooling and average pooling, followed by linear layer. Table 4.5 shows attention pooling performs 7.9% higher than average pooling on the ImageNet classification task. For attention pooling, we keep the embedding dimension the same as the intermediate feature dimensions.

Input-Target	Tokens	Vocabulary	Top1
VQGAN-VQGAN	16x16	16k	61.3
VQGAN-VQGAN	16x16	1k	61.1
dVAE-dVAE	32x32	8k	61.2
dVAE-dVAE	16x16	8k	53.2
patch-patch	16x16	-	60.6
patch-dVAE	16x16	8k	58.5

Table 4.3: **ImageNet Linear Probing Accuracy with Various Tokenizers:** We compare discrete (dVAE, VQGAN) and patch embedding as input and target for pre-training our models. ImageNet top-1 accuracies are computed by linear probing at the 9th layer of the `large` model.

Method	Compute	Top1
dVAE/16	1.42×10^{17}	53.2
dVAE/32	5.68×10^{17}	61.2
dVAE/16→32	2.13×10^{17}	63.2
dVAE/16→32†	2.13×10^{17}	64.4

Table 4.4: **Token Resolution:** While the performance is lower for a low-resolution model, when finetuned for next-patch prediction at a higher resolution, its performance surpasses the full-resolution pre-trained model. † Base values of the RoPE is 50,000.

Method	Tokens	Pooling	Top1
dVAE	16x16	Average	53.2
dVAE	16x16	Attention	61.1

Table 4.5: **Attention vs Average Pooling:** When probed at the same layers, attention pooling performs much better than average pooling of intermediate tokens.

Resolution: When training with dVAE tokens, a 256x256 image results in 1024 tokens, this is four times more number of tokens compared to patch embeddings or VQGAN tokens. If we reduce the number of tokens to 256, then the effective image resolution becomes 128x128. Table 4.4 shows a clear drop in performance when pre-training the model at 128x128 resolution. However, due to the use of relative positional embeddings (RoPE [126]), we can easily finetune the 128x128 (or 16x16 token equivalent) model for higher resolution. Surprisingly, this does better than pre-training at 256x256 resolution and requires only one epoch of finetuning. Not only does this improve the performance, but the pre-training also becomes cheaper compared to full-resolution pre-training. †Additionally, Fine-tuning with higher base values of the RoPE embeddings (50,000) leads to better accuracy.

Architecture: We train various language models from GPT2 [103] with absolute sine-cosine positional embeddings, and non-transformer based model Mamba [50] only using dVAE tokens. We

mimicked the GPT2 architecture and do architecture comparisons. We compare these models with LLaMA [131]. We evaluate linear probing performance at each layer of these models and report the best performance in Table 4.6.

Probing Layer: When probing the pre-trained models, especially the decoder-only model best performance is observed at the middle layers. This behavior is first observed in iGPT [20]. Figure 4.4 shows the peak performance on recognition occurs at about 50% of the depth of the model. This behavior holds across all model sizes. While in MAE [54] and BEiT [8] encoder-decoder models, due to the uneven nature of the encoder and decoder, the best features are observed at the top of the encoder layers. However, on decoder-only models with uniformly distributed layers, the last layers perform worse on recognition tasks, mainly because these layers are trained to reconstruct the input. More probing results with various tokenizers, resolutions, and probing methods are shown in the supplementary material.

Model	Params	Top1
GPT2 [103]	280 m	48.5
Mamba [50]	290 m	40.7
LLaMA [131]	280 m	53.2

Table 4.6: **Architecture:** We compare sequence modeling architectures LLaMA [131], GPT2 [103], and non-transformer models, Mamba [50] on ImageNet linear probing task.

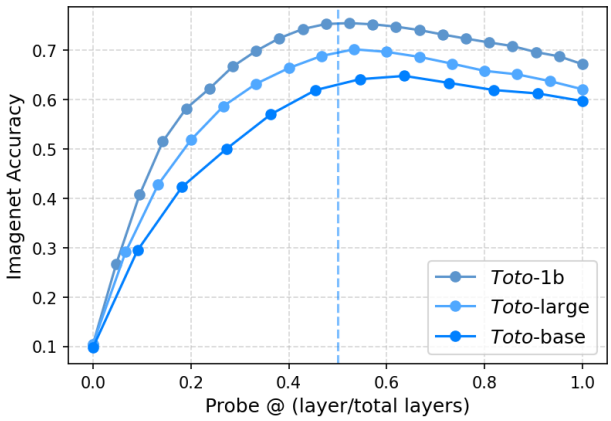


Figure 4.4: **Probing at Different Layers:** We show the attention-probing performance at each layer of our three models. Peak performance is observed at around 50% depth of the models.

Image Recognition

To measure the representation quality of our pre-trained models, we evaluate our models on ImageNet-1k [29] classification. We apply a probe at each layer of the model, with attention pooling, and choose the optimal layer with the highest classification accuracy. We fine-tune the pre-trained models further by applying self-supervised next token prediction loss in Eq 4.2, together with cross-entropy loss applied for probing layers (with stop-gradients). We train the probing layers for 90 epochs, with a learning rate of $6e^{-5}$. We also use layer decay of 0.9 to reduce the learning rate at the early layers of the model. During this stage, all the models are fine tuned with 32×32 token resolution, on the self-supervised loss, and increase the base value of the RoPE [126] embeddings from 10,000 to 50,000 support larger resolution.

Table 4.7 shows the ImageNet top-1 accuracy of our `base, large` and `1b` models. First,

there is a clear difference in terms of classification performance when it comes to discriminative models versus generative models. Instance discriminative models such as SimCLR [21], and DINO [16] are trained to separate samples from each other and they are designed to perform well on discriminative tasks. On the other hand, generative models are *just* trying to model the data distribution. While achieving comparable performance to other generative models on image recognition, among autoregressive generative models, our model achieved the highest top-1 accuracy. The scaling of data, and the use of tokens instead of pixels, allows our one billion parameter model to achieve similar performance compared to iGPT [20] 7 billion models.

Method	**Arch**	**#θ**	**Top1**
Discriminative Approaches			
SimCLR [21]†	RN50x2	94	74.2
BYOL [49]†	RN50x2	94	77.4
SwAV [17]†	RN50x2	94	73.5
DINO [16]	ViT-B/8	86	80.1
DINOv2 [96]	ViT-g/14	1011	86.4
Generative Approaches			
BEiT-L [8]	ViT-L/14	307	62.2
AIM [94]	ViT-1B/14	1200	80.6
MAE [54]	ViT-H/14	632	80.9
iGPT-L [20]†	GPT-2	1386	65.2
iGPT-XL [20]†	GPT-2	6801	72.0
Toto-base	LLaMA	120	64.7
Toto-large	LLaMA	280	71.1
Toto-1b	LLaMA	1100	75.3

Table 4.7: **ImageNet Results:** We compare discriminative and generative models on ImageNet [29] recognition task. While achieving comparable performance among generative models, our models model achieves the highest accuracy on autoregressive modeling. †models are evaluated with linear probing.

Action Recognition

We use Kinetics-400 (K400) [66] for evaluating our models on action recognition tasks. Similar to ImageNet evaluation, we apply a probe at each layer of the model, with attention pooling, and choose the optimal layer with the highest action classification accuracy. We also fine-tune the pre-trained models on a self-supervised next-patch prediction task while training the probing layers with a classification loss. All our video models are trained with 16 frames, thus with a context length of 4096 tokens per video. When evaluating videos, we follow the protocol in SlowFast [41]. Unlike ImageNet where we evaluate the models at 256x256 resolution, on videos we only evaluate our models at 128x128 resolution, to keep the number of tokens in a similar budget.

Table 4.8 shows the Kinetics-400 top-1 accuracy of our `base, large` and `1b` models. Similar to ImageNet results in Table 4.7, we see that discriminately trained models perform better than generative models. Our models achieve comparable performance among generative models, and first to show competitive performance on action recognition with autoregressive generative modeling. All the models are trained and evaluated with 16 frames with a stride of 4 frames.

Method	**Arch**	**Top1**
Discriminative Approaches		
I-JEPA [5]	ViT-H/16	74.5
OpenCLIP [22]	ViT-G/14	83.3
DINOv2 [96]	ViT-g/14	84.4
InternVideo [143]	-	73.7
Generative Approaches		
Hiera [118]	Hiera-H/14	77.0
MVD [140]	ViT-H/14	79.4
VideoMAE [139]	ViT-L/14	79.8
Toto-base	LLaMA	59.3
Toto-large	LLaMA	65.3
Toto-1b	LLaMA	74.4

Table 4.8: **K400 Results:** We compare discriminative and generative models on Kinetics-400 [66] action recognition task. While achieving comparable performance among generative models, our models are the first to show the competitive performance on K400 with autoregressive pre-training, and shows scaling with large model sizes.

Action Forecasting

While the Kinetics dataset captures internet-style exocentric videos, Ego4D [48] videos capture day-to-day life egocentric videos. A general vision model should be able to reason about both exo and ego-centric videos. Task-wise, Kinetics requires the model to reason about the action using full context (e.g. the model has seen the action), while the Ego4D short-term action anticipation v1 task requires models to predict future actions from past context. We use our models as the backbone for the pyramid network used in StillFast [107] extract tokens at 5 layers and fuse them with the pyramid network. We fully fine-tuned our model with self-supervised next-patch loss along with task-related losses, and we observed having self-supervision loss improves overall performance. Table 4.9 shows the performance of our `large` model on the Ego4D short-term action anticipation task. This task requires predicting the object to be interacted with (noun) and the type of interaction (verb) as well as time to contact (ttc) from the last seen frame to an estimated time between object-hand contact. As shown in Table 4.9, these tasks are difficult with maximum overall mean-average precision of 2.70.

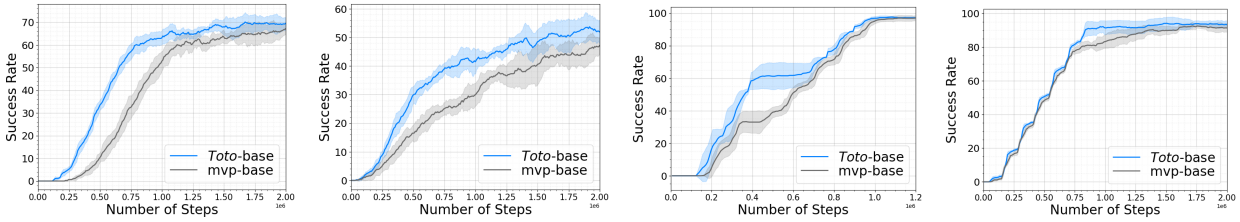
Method	**Noun**	**N+V**	**N+TTC**	**Overall**
FRCNN+Rnd [48]	17.55	1.56	3.21	0.34
FRCNN+SF [48]	17.55	5.19	5.37	2.07
Hiera-large [118]	14.05	6.03	4.53	2.12
StillFast [107]	16.20	7.47	4.94	2.48
VideoMAE-large [139]	15.16	6.72	5.26	2.55
MAE-ST-large [40]	13.71	6.63	4.94	2.60
Toto-large	15.20	6.75	5.41	2.70

Table 4.9: **Ego4D Results:** Our model achieves comparable mean-average precision compared to previous work. We compare our method with, FRCNN+Rnd [48], FRCNN+SF [48], Hiera [118], StillFast [107], VideoMAE [139], and MAE-ST [40].

Video Tracking

We study our pre-trained models on label propagation using the protocols in [60] on DAVIS dataset [101]. Compared to previous tasks such as classification, and forecasting, this evaluation does not require finetuning or probing of the features. Following [60], we use the features from the last n frames to find the nearest neighbor patch in the current frame, and then propagate the masks from the previous frames to the current frame.Comparison with Dino [16] and MAE [54] is shown in Table 4.10 and qualitative results are shown in Figure 4.5.

Figure 4.5: **Semi-Supervised Tracking:** We follow the protocol in STC [60], start with the GT segmentation mask, and propagate the labels using the features computed by *Toto*-large. The mask was propagated up to 60 frames without losing much information.



(a) Franka Pick (b) Kuka Pick (c) Franka Cabinet (d) Kuka Cabinet

Figure 4.6: **Robot Manipulation with Reinforcement Learning:** We compare MAE-base [105] with *Toto*-base pre-trained models in simulation following [154]. We evaluate each model the mean success rate over training steps. *Toto* was able to learn these tasks faster than MAE, across two robots and two tasks.

Robotics

In this section, we study the effectiveness of our pre-trained representations for robotic manipulation. We consider tasks in both simulation and in the real world. Real world experiments needs to run at real time, there for we only use *Toto*-base models, in both setting. Despite being a small model, *Toto*-base can achieve better performance in simulation and on-par performance to state-of-the-art robot models in real world experiments.

Simulation Experiments: Following the protocols in MVP [154], we use our visual pre-trained models to embed pixel observations. The model is frozen and we only take tokens at an intermediate layer, apply average pooling, and learn the linear layer on top to embed pixel observations. These observations are used to train DAgger policies for 4 different tasks: Franka-pick 4.6a, Kuka-pick 4.6b, Franka-cabinet 4.6c, and Kuka-cabinet tasks 4.6d. Figure 4.6 shows the mean success rate over training steps. Compared to the MVP baseline, our model was able to learn these tasks

Method (Res/Patch)	J&F	J	F
DINO-base (224/8)	54.3	52.5	56.1
DINO-base (224/16)	33.1	36.2	30.1
MAE-base (224/16)	31.5	34.1	28.9
Toto-base (256/8)	42.0	41.2	43.1
Toto-large (256/8)	44.8	44.4	45.1
Toto-1b (256/8)	46.1	45.8	46.4
Toto-large (512/8)	62.4	59.2	65.6

Table 4.10: **DAVIS Tracking:** We report J, F, and J&F scores at the peak layers of each model. We achieves comparable performance as DINO and at large resolution (512), it outperforms all methods.

Model	# Traj	Success
MVP	240	75%
Toto-base	240	63%

Table 4.11: **Robotics, Real-world Experiments:** We compare MVP [105] and *Toto* on a Franka cube-picking task in the real world. Features from both models are pre-trained, frozen, and passed into a learning module trained with behavior cloning using the same demonstrations. We see that our approach performs comparably to the state-of-the-art vision backbone for robotics, despite not being designed with the robotic application in mind.

faster with better sample efficiency across robots and tasks. For fair comparisons, we use the best MAE model from MVP [105] which is trained on ImageNet [29], Ego4D [48] and 100DOH [120] datasets.

Real-world Experiments: Next, we evaluate our pre-trained representations in the real world. We follow the setup from [105]. We extract vision features using a pre-trained vision encoder and train a controller on top of frozen representations using behavior cloning. Specifically, we consider a cube picking tasks using a 7 DoF Franka robot, shown in Figure 4.7. We use the demonstrations provided by [106]. In Table 4.11 we compare our model to a vision encoder from [105]. We report the success rate over 16 trials with variations in object position and orientation. Our model performs favorably to a vision encoder pre-trained for robotics.

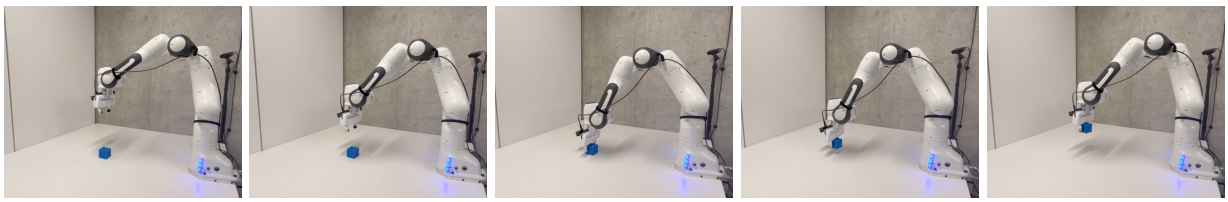


Figure 4.7: **Real-world Deployment:** We show an example episode of our policy performing the cube picking task on a Franka robot in the real world. We use *Toto*-base to run the robot at real time, despite being a small model, *Toto* was able to achieve about 63% success rate in real world setting.

Method	Model	16	32
V3D	ResNet	55.2	69.7
TFC V3D	ResNet	54.6	70.2
Toto-large	LLaMa	62.8	72.9

Table 4.12: **Object Permanence:** CATER [44] object localization task, where the object is hidden under or obstructed by other objects. The model is trained to predict its coarse location. Our model performs better than previous methods on snitch localization task at 16, 32 temporal resolutions.

Object Permanence

To quantitatively measure the performance of how well the model understands object permanence, we evaluate our models on CATER localization task [44]. Here, a ball is moving in the scene, and the task is to find its location in the 6 by 6 grid. We fine tune our *Toto*-large model on this task at temporal resolutions 16, and 32 frames. In both cases, our pre-trained models were better at localizing the target compared to models trained specifically for this task, such as V3D [164], TFC-V3D [164]. Table 4.12 shows the performance on the CATER snitch localization task, and *Toto*-large achieve 62.8% and 70.9% performance with 16 and 32 frames respectively.

Probing Across Layers

As shown in Figure 4.4 for the ImageNet classification task, different layers of the model contribute to the task differently for the image classification task; this behavior is also observed in iGPT [20]. To study this behavior across multiple tasks, we train probing layers for action recognition, object tracking, and robot manipulation. Figure 4.8 shows probing performance across layers, model size, and tasks. It shows that action recognition follows a similar trend to ImageNet classification, having peak performance at the middle of the model stacks. While Object tracking also shares a similar trend with image classification and action recognition, object manipulation shows an interesting trend of the last layers performing well as middle layers from picking objects. Compared to the first three tasks, robot manipulation has a generative nature as a task and can benefit from generative

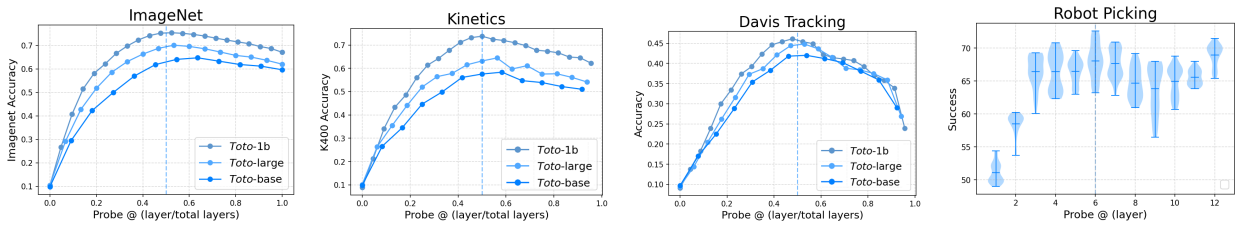


Figure 4.8: **Probing Across Layers, Models, and Tasks:** We study the behavior of our models across multiple layers and tasks. For image classification, action recognition, and object tracking, all the models behave similarly and peak around 50% of the model depth. This behavior is observed across all model sizes. Robot tasks show a similar behaviour, where the middle layers perform good at picking the objects, but last layers also perform good as middle layers. These plots suggests, in decoder-only model, first half of the model starts to behave like an encoder, and compress the information, and then rest of the model, projects the compressed semantic features back to input space.

pre-training. In encoder models [16] or encoder-decoder models [54, 8] the last layer of the encoder has more semantic features. *This may suggest that, in decoder-only model, first half of the model starts to behave like an encoder, and compress the information, and then rest of the model, projects the compressed semantic features back to input space.*

Compute Optimal Scaling

We study the scaling behaviors of *Toto* using μ-Parameterization [159]. First we train various models, a1-a6, with linearly increasing hidden size and number of layers (Table 4.15). All models use the VQGAN tokenizer [34]. We then optimize the learning rate for these models, with μ-Parameterization [159]. Figure 4.11 shows optimal learning rate of 2^{-7} for all of the model widths. Once we find the optimal learning rate, we train a1-a6 models on our data mixture (Table 4.2). Figure 4.9 shows the loss vs training compute of *Toto* models. This shows a clear power law relationship between the compute and validation loss. Based on these experiments *Toto* shows a power law of $L(C) = 7.32 \cdot C^{-0.0378}$. For comparison, the GPT-3 power law relationship [14] is $L(C) = 2.57 \cdot C^{-0.048}$. While these are not comparable directly, the scaling coefficients indicate how much change in loss to expect for extra added compute. This suggests that the visual next token prediction models, such as *Toto*, scale but at a slower rate than language models.

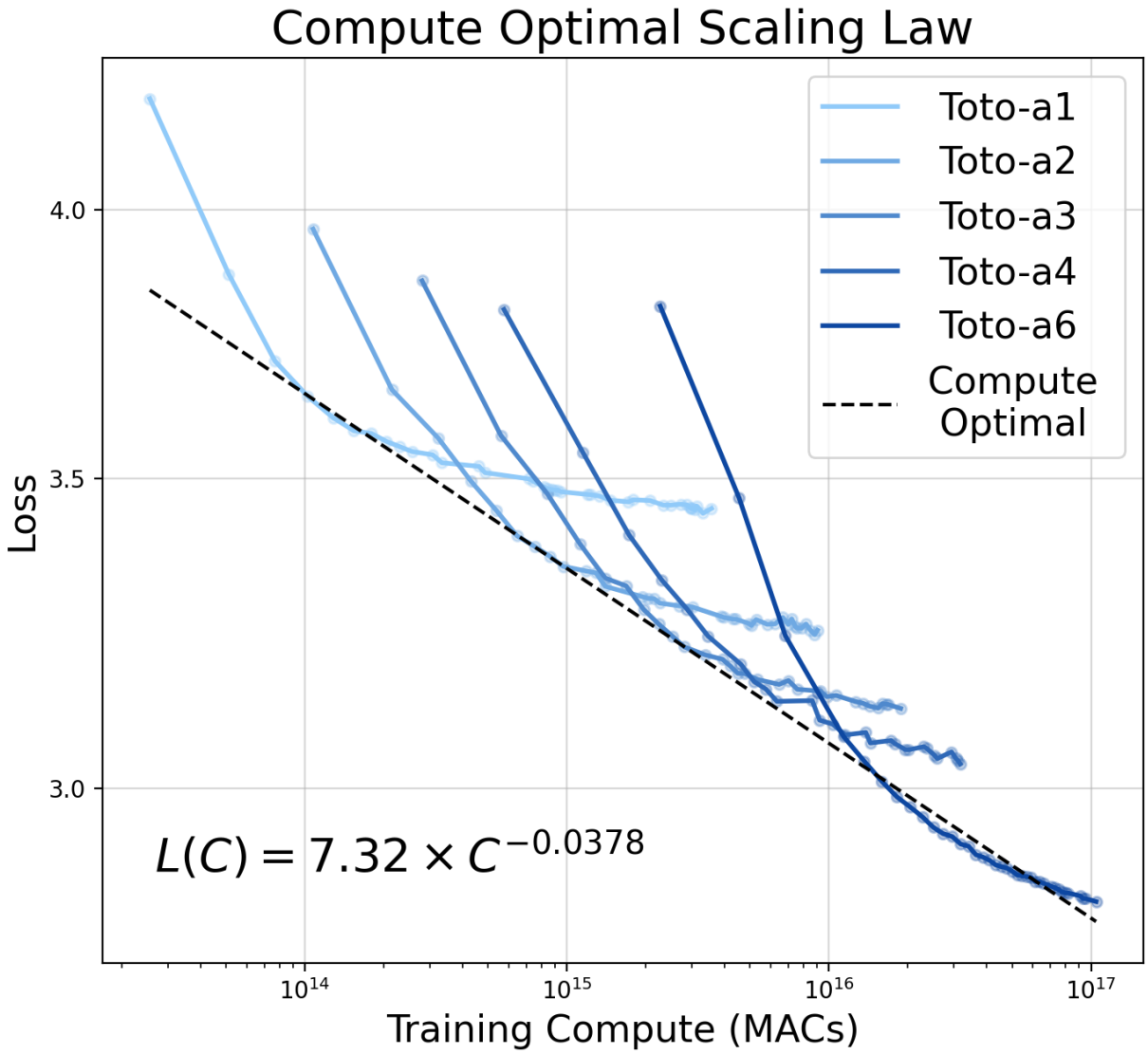


Figure 4.9: **Scaling *Toto*:** We train multiple variants of *Toto*, with increasing hidden size and depth, with optimal learning rates. We plot the validation loss vs the compute spent on training in MACs. This shows a clear scaling behavior with optimal compute.

4.5 Limitations

Our study suggests several important limitations and opportunities for future work. A significant limitation stems from the use of internet videos, which, unlike carefully curated datasets, introduces challenges related to data quality and diversity. This variance in data quality can impact model performance, especially when compared to models trained on more curated datasets. Another limitation is the use of tokenizer, this makes the learning not end-to-end, and the representation and generation quality is bounded by the quality of the tokenizer, and with quantized vectors, the quality is very much limited, this needs further explorations to build a universal visual tokenizer. Another fundamental limitation is training on videos for next token prediction task. The added redundancy in video frames, can hurt quality of the learned representations. See Appendix 4.8 for more discussion on this topic. Additionally, our exploration of various design choices are based on ImageNet classification. While it does transfer to most of the tasks we considered in this chapter, it may not be the optimal configuration for many other tasks. Furthermore, we have not yet fully assessed our method's effectiveness in dealing with dense prediction tasks, fine-grained recognition, or comprehending complex temporal dynamics over extended time frames. These areas represent key opportunities for further research, aiming to broaden the fruitfulness of autoregressive pre-trained models.

4.6 Conclusion

We empirically studied autoregressive pre-training from images and videos. We curated a large video dataset and conducted a large-scale evaluation across a range of diverse tasks, including image recognition, video classification, video forecasting, object tracking, object permanence, and robotic manipulation. We performed extensive ablation studies to understand different design choices and compared auto regressive pre-training from videos to strong baselines across different tasks. We found that, despite minimal inductive biases, our approach achieves competitive performance across all tasks. Finally, we studied the scaling behavior of visual next token prediction models, and showed it scales with compute, but at a slower rate than text based next token prediction models.

4.7 Acknowledgments

We thank Andrea Madotto, Po-Yao (Bernie) Huang, and Shiry Ginosar for helpful discussions. We're grateful to Ronghang Hu and Xinlei Chen for their help with TPU setup and code bases. We also thank Baifeng Shi for helping us with robots evaluations. We thank Valentin Gabeur and Neerja Thakkar for their valuable feedback on the chapter.

4.8 Additional Details

Video Tokens for Pre-Training

The next patch prediction for visual pre-training is equivalent to the next token prediction in large language models. However, most languages have a clear sequential nature, therefore there is a clear definition for the next word. This also makes the next word prediction task relatively harder, since the model requires learning to extrapolate the data. On the other hand, images and videos, especially over the spatial dimensions lack a sequential nature. We follow the previous works [20, 134] to make the images and videos into a 1D sequence by scanning the patches in raster order. While this ordering allows for example to learn to predict the bottom half of the image from the top part of the image, in many places, the tokens can be predicted by interpolating rather than extrapolating.

On the time axis, yes, there is a clear sequential nature, however, video frames compared to text tokens are more redundant, making the next frame prediction task much easier. Figure 4.10 shows average validation loss over 4096 token, in kinetics 400 dataset [66], on *Toto*-large model. This shows there is high loss of the first frame, but the subsequent frames have relatively lower loss compared to the first frame. This is because, even with reasonably lower sampling rate, frames following the first frame has some redundancy, and hinders the learning, since these tokens are relatively easy to predict. This also could be attributed by emergence of induction heads [95]. While we focused on learning from unfiltered internet scale video with minimal inductive bias, to learn efficiently from videos, need further research in this direction.

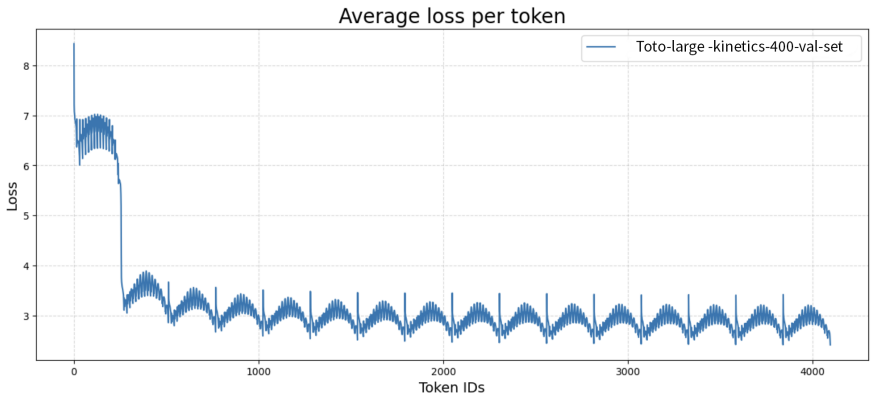


Figure 4.10: **Average Validation Loss Over Tokens:** We show the average loss per token for kinetics validation set. It clearly shows the redundancy in videos, as the first frame has higher prediction loss, and rest of the frames on average has lower loss than the first frame.

Prefix attention

During fine-tuning, we experimented with causal and full attention. On ImageNet, our base model achieved full attn: 82.6% vs causal attn: 82.2%. Even though our models are *not pre-trained with prefix attention*, still able to utilize full attn at fine-tuning. This is an unrealized benefit of training with videos, (a middle token in say, 8th frame won't see the rest half of the 8th frame, but have seen all the tokens from 7th frame, which are similar because of video, hence approximating full attention at pre-training)

Full fine-tuning

We fine-tuned our models on ImageNet, and performance is close to SOTA, compared to linear probing (where we only use causal attention). But during the fine-tuning, we use full attention.

DINO	MoCo v3	BEiT	MAE	*Toto*
82.8	83.2	83.2	83.6	82.6

Table 4.13: **Full Fine Tuning Performance:** Comparison of different methods performance on ImageNet-1K.

iGPT vs *Toto* on ImagenNet

Table 4.7 shows ImageNet evaluation performance. However, iGPT [20] models are evaluated only using linear probing. To have a fair comparison, between iGPT and *Toto*, we reevaluated our models using linear probing. Both models have causal attention and are trained on auto-regressive objectives. On the same model sizes, about 1 billion parameters, our achieve 66.2% while the similar iGPT model's ImageNet performance is 65.2%. This fair evaluation suggests the modifications made on *Toto* have clear benefits over iGPT.

Method	Arch	#θ	Top1
iGPT-L [20]	GPT-2	1386	65.2
Toto-1b	LLaMA	1100	66.2

Table 4.14: **ImageNet Linear Probing Results:** *Toto* performs better than similar size iGPT models.

μ-Parameterization

To study the scaling behaviours of *Toto* using μ-Parameterization [159]. First we train various models a1-a6 (in Table 4.15), with hidden sizes (64-1536) and number of layers (12-48), increasing linearly and we used VQGAN tokenizer [34]. Then we tune the learning rate for these models, with fixed depth using μ-Parameterization [159]. Figure 4.11 shows optimal learning rate of 2^{-7} for all the model widths. Once we find the optimal learning rate, we train a1-a6 models on the mixture of image and video data, as mentioned in Table 4.2.

Model	Params	Dimension	Heads	Layers
a1	14.8M	256	16	12
a2	77.2M	512	16	16
a3	215M	768	16	20
a4	458M	1024	16	24
a5	1.2B	1536	16	28
a6	1.9B	1792	16	32

Table 4.15: ***Toto* Varients:** We scale *Toto* models by increasing hidden dimension and number of layers linearly while keeping number of heads constant following [159, 131].

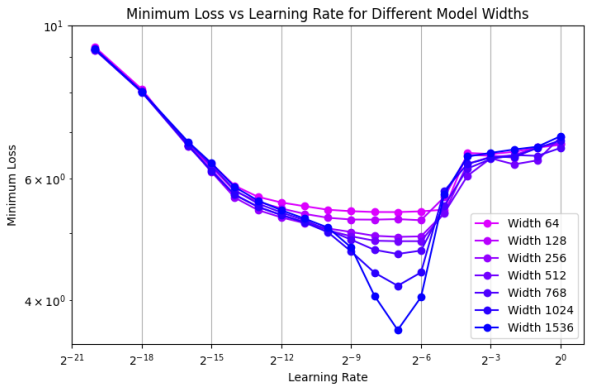


Figure 4.11: **μ-Parameterization Learning Rate:** We show that μ-Parameterization [159], we can train all width *Toto* models, with an single optimal learning rate of 2^{-7}.

n-gram distribution

In this section, we compare the 2-gram and 3-gram distribution of dVAE [112], VQGAN [34] image tokeizers. We compute 2-gram and 3-gram distributions on the discrete tokens of 10000 ImageNet validation images. Figure 4.12 and Figure 4.13 show the distributions of these tokenizers respectively. On 2-gram distribution, dVAE [112] has more discrete combination of tokens compared to both VQGAN-1K and VQGAN-16k tokenizers.

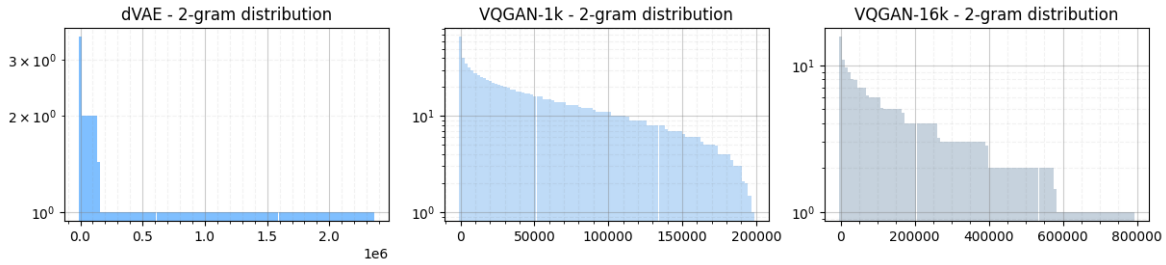


Figure 4.12: **2-gram Distribution of Various Tokens:** We compute the 2-gram distribution on 10000 images from the ImageNet validation set. Compared to VQGAN 1k and 16k vocabulary tokenizers, the dVAE tokenizer has a larger set of token combinations.

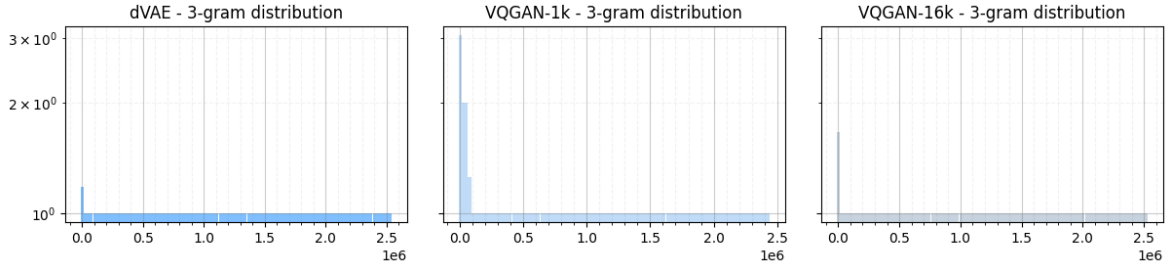


Figure 4.13: **3-gram Distribution of Various Tokens:** We compute the 3-gram distribution on 10000 images from the ImageNet validation set. All the tokenizers has similar almost flat distribution when it comes to 3-gram tokens.

Attention probing variants on K400

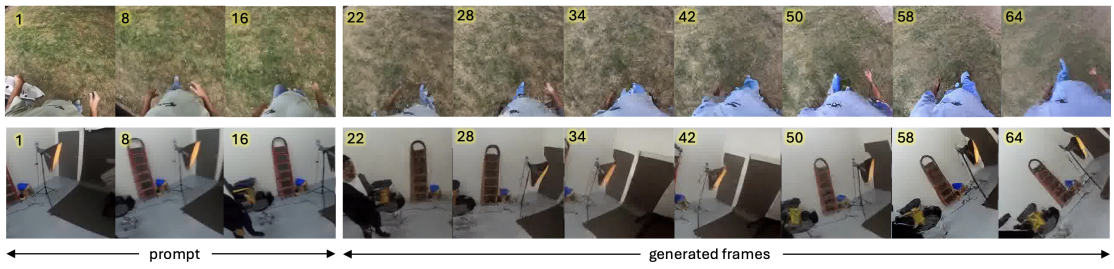
We also evaluate our models and baselines on the Kinetics 400 dataset using a variant of attention probing. In the main chapter, we use attention probing, with only learning W_k, W_v matrices, and a single learnable query vector. We also test with cross attention with MLP layers as the attention classifier, to give more capacity to the learnable head. Table 4.16 show the performance on the attention classifier with an additional MLP head. This helps to improve performance across over all models.

Method	Arch	Top1
Hiera [118]	Hiera-L/14	74.2
Hiera [118]	Hiera-H/14	75.2
VideoMAE [139]	ViT-B/14	65.4
VideoMAE [139]	ViT-L/14	74.8
Toto-base	LLaMA	61.2
Toto-large	LLaMA	65.8
Toto-1b	LLaMA	74.8

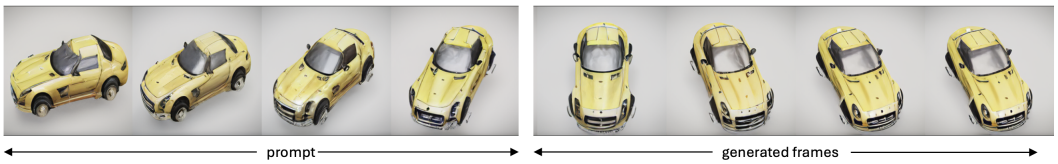
Table 4.16: **K400 Results:** We evaluate our models using cross attention and MLP layer as the classification head. Overall using a high-capacity head improves the performance across all models.

Generation samples

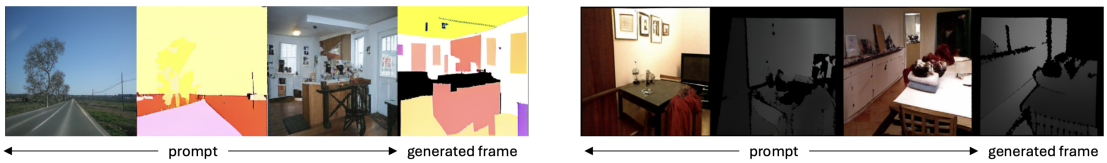
long video generation: we can generate up to 64 frames, first raw: periodic motion, second raw: object permanence (light stand).



prompting (pre-trained model): shows 3D rotation



prompting (finetuned model): A small 1000-step fine-tuning leads to a promptable model for various vision tasks.



Additional Layer-wise Probing Results

We probe the multiple variants of our models at each layer for the best ImageNet performance. First, we test the models on linear probing, on both sizes of 128 and 256 resolution. Figure 4.14 presents the probing curves of the models trained with attention probing at 128 resolution. Across all models, the performance has a similar behavior to the pre-trained models, with peak performance around the middle of the depth of the model.

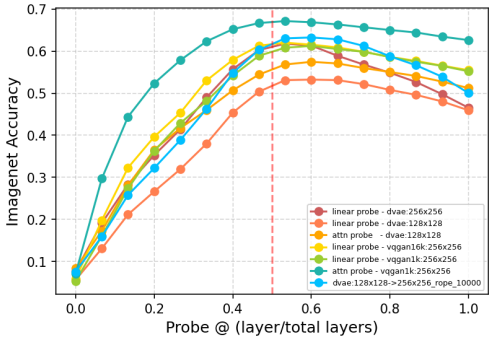


Figure 4.14: **Training Loss Curves:** We show the training loss curves for multiple variants of our models.

Chapter 5

Conclusion

This thesis has explored how to build temporally coherent, human-centered video models—models that not only see but follow, recognize, and predict. From the fundamental problem of tracking people in monocular video, to action recognition using structured 3D trajectories, to large-scale self-supervised learning from raw pixels, the work presented here takes a layered approach to understanding video through the lens of people and motion.

In the first part, we introduced PHALP, a 3D tracking framework that predicts a person's appearance, location, and pose over time. By lifting detections into 3D and aggregating them into predictive tracklets, PHALP handles occlusions, scene changes, and complex motion with stability. It shows that 3D representations—despite being estimated from monocular video—enable more robust and interpretable tracking than traditional 2D methods.

Building on this, the second part proposed LART, a Lagrangian approach to action recognition. By treating people as entities evolving in space-time and fusing their pose and appearance over trajectories, LART offers a more natural and effective way to classify human actions—especially those involving interactions or temporally extended behavior. The model highlights the importance of moving beyond static or grid-based video tokenization and instead focusing on structured, person-centric representations.

The final part introduced Toto, a family of autoregressive video transformers trained on a massive scale with minimal supervision. Toto models learn to predict the next token in videos. Despite its simplicity, this objective yields versatile models that perform competitively across diverse tasks like classification, tracking, and even reasoning about object permanence. Furthermore, it shows of power-law scaling behavior offers insights into how video model performance grows with compute and data, paving the way for more general-purpose visual learners.

Ultimately, the goal is to understand the world through video— to trace the story of people and pixels over time. My hope is that this thesis brings us one step closer to building systems that do that.

Bibliography

[1] Emre Aksan et al. “A spatio-temporal transformer for 3D human motion prediction”. In: *3DV*. 2020.

[2] Alexandre Alahi et al. “Social LSTM: Human trajectory prediction in crowded spaces”. In: *CVPR*. 2016.

[3] Mykhaylo Andriluka et al. “PoseTrack: A benchmark for human pose estimation and tracking”. In: *CVPR*. 2018.

[4] Anurag Arnab et al. “ViViT: A video vision transformer”. In: *ICCV*. 2021.

[5] Mahmoud Assran et al. “Self-supervised learning from images with a joint-embedding predictive architecture”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 15619–15629.

[6] Fred Attneave. “Some informational aspects of visual perception.” In: *Psychological review* (1954).

[7] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. “Layer normalization”. In: *arXiv preprint arXiv:1607.06450* (2016).

[8] Hangbo Bao et al. “Beit: Bert pre-training of image transformers”. In: *arXiv preprint arXiv:2106.08254* (2021).

[9] Fabien Baradel et al. “Leveraging MoCap Data for Human Mesh Recovery”. In: *3DV*. 2021.

[10] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. “Tracking without bells and whistles”. In: *ICCV*. 2019.

[11] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. “Is space-time attention all you need for video understanding?” In: *ICML*. 2021.

[12] Federica Bogo et al. “Keep it SMPL: Automatic estimation of 3D human pose and shape from a single image”. In: *ECCV*. 2016.

[13] Christoph Bregler and Jitendra Malik. “Tracking people with twists and exponential maps”. In: *CVPR*. 1998.

[14] Tom B Brown. “Language models are few-shot learners”. In: *arXiv preprint arXiv:2005.14165* (2020).

[15] Tom B Brown et al. “Language models are few-shot learners”. In: (2020).

[16] Mathilde Caron et al. “Emerging properties in self-supervised vision transformers”. In: *ICCV*. 2021.

[17] Mathilde Caron et al. “Unsupervised learning of visual features by contrasting cluster assignments”. In: *Advances in neural information processing systems* 33 (2020), pp. 9912–9924.

[18] Joao Carreira and Andrew Zisserman. “Quo vadis, action recognition? a new model and the kinetics dataset”. In: *CVPR*. 2017.

[19] Joao Carreira et al. “A short note on the kinetics-700 human action dataset”. In: *arXiv preprint arXiv:1907.06987* (2019).

[20] Mark Chen et al. “Generative pretraining from pixels”. In: *International conference on machine learning*. PMLR. 2020, pp. 1691–1703.

[21] Ting Chen et al. “A simple framework for contrastive learning of visual representations”. In: *International conference on machine learning*. PMLR. 2020, pp. 1597–1607.

[22] Mehdi Cherti et al. “Reproducible scaling laws for contrastive language-image learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 2818–2829.

[23] Vasileios Choutas et al. “Accurate 3D Body Shape Regression Using Metric and Semantic Attributes”. In: *CVPR*. 2022.

[24] Vasileios Choutas et al. “PoTion: Pose motion representation for action recognition”. In: *CVPR*. 2018.

[25] Gioele Ciaparrone et al. “Deep learning in video multi-object tracking: A survey”. In: *Neurocomputing* 381 (2020), pp. 61–88.

[26] Kevin Clark et al. “Electra: Pre-training text encoders as discriminators rather than generators”. In: *arXiv preprint arXiv:2003.10555* (2020).

[27] R James Cotton. “PosePipe: Open-Source Human Pose Estimation Pipeline for Clinical Research”. In: *arXiv preprint arXiv:2203.08792* (2022).

[28] Patrick Dendorfer et al. “MOTChallenge: A benchmark for single-camera multiple target tracking”. In: *IJCV* 129.4 (2021), pp. 845–881.

[29] Jia Deng et al. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[30] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: 2019.

[31] Piotr Dollár et al. “Behavior recognition via sparse spatio-temporal features”. In: *2005 IEEE international workshop on visual surveillance and performance evaluation of tracking and surveillance*. 2005.

[32] Alexey Dosovitskiy et al. “An image is worth 16x16 words: Transformers for image recognition at scale”. In: (2021).

[33] Alexei A Efros et al. “Recognizing action at a distance”. In: *ICCV*. 2003.

[34] Patrick Esser, Robin Rombach, and Björn Ommer. “Taming transformers for high-resolution image synthesis. 2021 IEEE”. In: *CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 12868–12878.

[35] Haoqi Fan et al. “Multiscale vision transformers”. In: *ICCV*. 2021.

[36] Alex Fang et al. “Data filtering networks”. In: *arXiv preprint arXiv:2309.17425* (2023).

[37] Hao-Shu Fang et al. “RMPE: Regional multi-person pose estimation”. In: *ICCV*. 2017.

[38] Christoph Feichtenhofer. “X3D: Expanding architectures for efficient video recognition”. In: *CVPR*. 2020.

[39] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. “Detect to track and track to detect”. In: *ICCV*. 2017.

[40] Christoph Feichtenhofer et al. “Masked Autoencoders As Spatiotemporal Learners”. In: *NeurIPS*. 2022.

[41] Christoph Feichtenhofer et al. “Slowfast networks for video recognition”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 6202–6211.

[42] Yutong Feng et al. “Relation modeling in spatio-temporal action localization”. In: *arXiv preprint arXiv:2106.08061* (2021).

[43] Golnaz Ghiasi et al. “Simple copy-paste is a strong data augmentation method for instance segmentation”. In: *CVPR*. 2021.

[44] Rohit Girdhar and Deva Ramanan. “CATER: A diagnostic dataset for Compositional Actions and TEmporal Reasoning”. In: *arXiv preprint arXiv:1910.04744* (2019).

[45] Rohit Girdhar et al. “Detect-and-track: Efficient pose estimation in videos”. In: *CVPR*. 2018.

[46] Georgia Gkioxari and Jitendra Malik. “Finding action tubes”. In: *CVPR*. 2015.

[47] Shubham Goel et al. “Humans in 4D: Reconstructing and Tracking Humans with Transformers”. In: *arXiv preprint (forthcoming)* (2023).

[48] Kristen Grauman et al. “Ego4d: Around the world in 3,000 hours of egocentric video”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 18995–19012.

[49] Jean-Bastien Grill et al. “Bootstrap your own latent-a new approach to self-supervised learning”. In: *Advances in neural information processing systems* 33 (2020), pp. 21271–21284.

[50] Albert Gu and Tri Dao. “Mamba: Linear-time sequence modeling with selective state spaces”. In: *arXiv preprint arXiv:2312.00752* (2023).

[51] Chunhui Gu et al. “AVA: A video dataset of spatio-temporally localized atomic visual actions”. In: *CVPR*. 2018.

[52] Peng Guan et al. “Estimating human shape and pose from a single image”. In: *ICCV*. 2009.

[53] Kaiming He et al. “Mask R-CNN”. In: *ICCV*. 2017.

[54] Kaiming He et al. “Masked autoencoders are scalable vision learners”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 16000–16009.

[55] Kaiming He et al. “Momentum contrast for unsupervised visual representation learning”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 9729–9738.

[56] Tom Henighan et al. “Scaling laws for autoregressive generative modeling”. In: *arXiv preprint arXiv:2010.14701* (2020).

[57] Gao Huang et al. “Deep networks with stochastic depth”. In: *ECCV*. 2016.

[58] Qingqiu Huang et al. “MovieNet: A holistic dataset for movie understanding”. In: *ECCV*. 2020.

[59] Catalin Ionescu et al. “Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments”. In: *PAMI* (2013).

[60] Allan Jabri, Andrew Owens, and Alexei Efros. “Space-time correspondence as a contrastive random walk”. In: *Advances in neural information processing systems* 33 (2020), pp. 19545–19560.

[61] Gunnar Johansson. “Visual perception of biological motion and a model for its analysis”. In: *Perception & psychophysics* 14.2 (1973), pp. 201–211.

[62] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. “Perceptual losses for real-time style transfer and super-resolution”. In: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*. Springer. 2016, pp. 694–711.

[63] Angjoo Kanazawa et al. “End-to-end recovery of human shape and pose”. In: *CVPR*. 2018.

[64] Angjoo Kanazawa et al. “Learning 3D human dynamics from video”. In: *CVPR*. 2019.

[65] Rangachar Kasturi et al. “Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol”. In: *PAMI* (2008).

[66] Will Kay et al. “The kinetics human action video dataset”. In: *arXiv preprint arXiv:1705.06950* (2017).

[67] Machiel Keestra. “Understanding human action. Integrating meanings, mechanisms, causes, and contexts.” In: (2015).

[68] Tarasha Khurana, Achal Dave, and Deva Ramanan. “Detecting invisible people”. In: *ICCV*. 2021.

[69] Alexander Kirillov et al. “PointRend: Image segmentation as rendering”. In: *CVPR*. 2020.

[70] Alexander Klaser, Marcin Marszałek, and Cordelia Schmid. “A spatio-temporal descriptor based on 3D-gradients”. In: *BMVC*. 2008.

[71] Muhammed Kocabas, Nikos Athanasiou, and Michael J Black. "VIBE: Video inference for human body pose and shape estimation". In: *CVPR*. 2020.

[72] Muhammed Kocabas et al. "PARE: Part Attention Regressor for 3D Human Body Estimation". In: *ICCV*. 2021.

[73] Muhammed Kocabas et al. "SPEC: Seeing People in the Wild with an Estimated Camera". In: *ICCV*. 2021.

[74] Jan J Koenderink. "Optic flow". In: *Vision research* 26.1 (1986), pp. 161–179.

[75] Nikos Kolotouros et al. "Learning to reconstruct 3D human pose and shape via model-fitting in the loop". In: *ICCV*. 2019.

[76] Nikos Kolotouros et al. "Probabilistic Modeling for Human Mesh Recovery". In: *ICCV*. 2021.

[77] Oh-Hun Kwon, Julian Tanke, and Juergen Gall. "Recursive Bayesian Filtering for Multiple Human Pose Tracking from Multiple Cameras". In: *ACCV*. 2020.

[78] Anders Boesen Lindbo Larsen et al. "Autoencoding beyond pixels using a learned similarity metric". In: *Proceedings of The 33rd International Conference on Machine Learning*. Ed. by Maria Florina Balcan and Kilian Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. New York, New York, USA: PMLR, 2016, pp. 1558–1566.

[79] Juho Lee et al. "Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks". In: *Proceedings of the 36th International Conference on Machine Learning*. Vol. 97. Proceedings of Machine Learning Research. PMLR, 2019, pp. 3744–3753.

[80] Ang Li et al. "The ava-kinetics localized human actions video dataset". In: *arXiv preprint arXiv:2005.00214* (2020).

[81] Yanghao Li et al. "MViTv2: Improved multiscale vision transformers for classification and detection". In: *CVPR*. 2022.

[82] Min Lin, Qiang Chen, and Shuicheng Yan. "Network in network". In: *arXiv preprint arXiv:1312.4400* (2013).

[83] Matthew Loper et al. "SMPL: A skinned multi-person linear model". In: *ACM Transactions on Graphics (TOG)* 34.6 (2015), pp. 1–16.

[84] Ilya Loshchilov and Frank Hutter. "Decoupled weight decay regularization". In: *arXiv preprint arXiv:1711.05101* (2017).

[85] Ilya Loshchilov and Frank Hutter. "Sgdr: Stochastic gradient descent with warm restarts". In: *ICLR*. 2017.

[86] Jonathon Luiten et al. "HOTA: A higher order metric for evaluating multi-object tracking". In: *IJCV* (2021).

[87] Dushyant Mehta et al. "Single-shot multi-person 3D pose estimation from monocular RGB". In: *3DV*. 2018.

[88] Dushyant Mehta et al. “XNect: Real-time multi-person 3D motion capture with a single RGB camera”. In: *ACM Transactions on Graphics (TOG)* 39.4 (2020), pp. 82–1.

[89] Tim Meinhardt et al. “TrackFormer: Multi-Object Tracking with Transformers”. In: *CVPR*. 2022.

[90] Tim Meinhardt et al. “TrackFormer: Multi-object tracking with transformers”. In: *CVPR*. 2022.

[91] Antoine Miech et al. “Howto100m: Learning a text-video embedding by watching hundred million narrated video clips”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 2630–2640.

[92] Daniel Neimark et al. “Video transformer network”. In: *ICCV*. 2021.

[93] John A Nelder and Roger Mead. “A simplex method for function minimization”. In: *The computer journal* 7.4 (1965), pp. 308–313.

[94] Alaaeldin El-Nouby et al. “Scalable Pre-training of Large Autoregressive Image Models”. In: *arXiv preprint arXiv:2401.08541* (2024).

[95] Catherine Olsson et al. “In-context learning and induction heads”. In: *arXiv preprint arXiv:2209.11895* (2022).

[96] Maxime Oquab et al. “Dinov2: Learning robust visual features without supervision”. In: *arXiv preprint arXiv:2304.07193* (2023).

[97] Junting Pan et al. “Actor-context-actor relation network for spatio-temporal action localization”. In: *CVPR*. 2021.

[98] Niki Parmar et al. “Image transformer”. In: *International conference on machine learning*. PMLR. 2018, pp. 4055–4064.

[99] Georgios Pavlakos, Jitendra Malik, and Angjoo Kanazawa. “Human Mesh Recovery from Multiple Shots”. In: *CVPR*. 2022.

[100] Georgios Pavlakos et al. “Expressive body capture: 3D hands, face, and body from a single image”. In: *CVPR*. 2019.

[101] Jordi Pont-Tuset et al. “The 2017 davis challenge on video object segmentation”. In: *arXiv preprint arXiv:1704.00675* (2017).

[102] Alec Radford et al. “Improving language understanding by generative pre-training”. In: (2018).

[103] Alec Radford et al. “Language models are unsupervised multitask learners”. In: (2019).

[104] Ilija Radosavovic et al. “Designing network design spaces”. In: *CVPR*. 2020.

[105] Ilija Radosavovic et al. “Real-world robot learning with masked visual pre-training”. In: *Conference on Robot Learning*. 2022.

[106] Ilija Radosavovic et al. “Robot Learning with Sensorimotor Pre-training”. In: *Conference on Robot Learning*. 2023.

[107] Francesco Ragusa, Giovanni Maria Farinella, and Antonino Furnari. "StillFast: An End-to-End Approach for Short-Term Object Interaction Anticipation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 3635–3644.

[108] Jathushan Rajasegaran et al. "An Empirical Study of Autoregressive Pre-training from Videos". In: *arXiv preprint arXiv:2501.05453* (2025).

[109] Jathushan Rajasegaran et al. "On the benefits of 3d pose and tracking for human action recognition". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 640–649.

[110] Jathushan Rajasegaran et al. "Tracking People by Predicting 3D Appearance, Location and Pose". In: *CVPR*. 2022.

[111] Jathushan Rajasegaran et al. "Tracking people with 3D representations". In: *NeurIPS*. 2021.

[112] Aditya Ramesh et al. "Zero-shot text-to-image generation". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 8821–8831.

[113] MarcAurelio Ranzato et al. "Video (language) modeling: a baseline for generative models of natural videos". In: *arXiv preprint arXiv:1412.6604* (2014).

[114] Davis Rempe et al. "HuMoR: 3D human motion model for robust pose estimation". In: *ICCV*. 2021.

[115] Ergys Ristani et al. "Performance measures and a data set for multi-target, multi-camera tracking". In: *ECCV*. 2016.

[116] Robin Rombach et al. "High-resolution image synthesis with latent diffusion models". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10684–10695.

[117] Olga Russakovsky et al. "Imagenet large scale visual recognition challenge". In: *International journal of computer vision* 115 (2015), pp. 211–252.

[118] Chaitanya Ryali et al. "Hiera: A Hierarchical Vision Transformer without the Bells-and-Whistles". In: *arXiv preprint arXiv:2306.00989* (2023).

[119] Anshul Shah et al. "Pose and Joint-Aware Action Recognition". In: *WACV*. 2022.

[120] Dandan Shan et al. "Understanding human hands in contact at internet scale". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 9869–9878.

[121] Claude E Shannon. "Prediction and entropy of printed English". In: *Bell system technical journal* (1951).

[122] Noam Shazeer. "Glu variants improve transformer". In: *arXiv preprint arXiv:2002.05202* (2020).

[123] Karen Simonyan and Andrew Zisserman. "Two-stream convolutional networks for action recognition in videos". In: *NIPS* (2014).

[124] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[125] Michael Snower et al. “15 Keypoints Is All You Need”. In: *CVPR*. 2020.

[126] Jianlin Su et al. “Roformer: Enhanced transformer with rotary position embedding”. In: *Neurocomputing* 568 (2024), p. 127063.

[127] Chen Sun et al. “Relational action forecasting”. In: *CVPR*. 2019.

[128] Graham W Taylor et al. “Convolutional learning of spatio-temporal features”. In: *ECCV*. 2010.

[129] Pavel Tokmakov et al. “Learning to Track with Object Permanence”. In: *ICCV*. 2021.

[130] Zhan Tong et al. “VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training”. In: *NeurIPS*. 2022.

[131] Hugo Touvron et al. “Llama: Open and efficient foundation language models”. In: *arXiv preprint arXiv:2302.13971* (2023).

[132] Du Tran et al. “Learning spatiotemporal features with 3D convolutional networks”. In: *ICCV*. 2015.

[133] Aaron Van den Oord et al. “Conditional image generation with pixelcnn decoders”. In: *Advances in neural information processing systems* 29 (2016).

[134] Aäron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. “Pixel recurrent neural networks”. In: *International conference on machine learning*. PMLR. 2016, pp. 1747–1756.

[135] Gül Varol et al. “Synthetic humans for action recognition from unseen viewpoints”. In: *International Journal of Computer Vision* 129.7 (2021), pp. 2264–2287.

[136] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).

[137] Heng Wang and Cordelia Schmid. “Action recognition with improved trajectories”. In: *ICCV*. 2013.

[138] Heng Wang et al. “Action Recognition by Dense Trajectories”. In: *CVPR*. 2011.

[139] Limin Wang et al. “Videomae v2: Scaling video masked autoencoders with dual masking”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 14549–14560.

[140] Rui Wang et al. “Masked Video Distillation: Rethinking Masked Feature Modeling for Self-supervised Video Representation Learning”. In: *CVPR*. 2023.

[141] Xiaolong Wang and Abhinav Gupta. “Videos as space-time region graphs”. In: *ECCV*. 2018.

[142] Xiaolong Wang et al. “Non-local neural networks”. In: *CVPR*. 2018.

[143] Yi Wang et al. “Internvideo: General video foundation models via generative and discriminative learning”. In: *arXiv preprint arXiv:2212.03191* (2022).

[144] Zelun Wang and Jyh-Charn Liu. "Translating math formula images to LaTeX sequences using deep neural networks with sequence-level training". In: *International Journal on Document Analysis and Recognition (IJDAR)* 24.1 (2021), pp. 63–75.

[145] Chen Wei et al. "Masked Feature Prediction for Self-Supervised Visual Pre-Training". In: *arXiv preprint arXiv:2112.09133* (2021).

[146] Chen Wei et al. "Masked feature prediction for self-supervised visual pre-training". In: *CVPR*. 2022.

[147] Philippe Weinzaepfel and Grégory Rogez. "Mimetics: Towards understanding human actions out of context". In: *IJCV* (2021).

[148] Dirk Weissenborn, Oscar Täckström, and Jakob Uszkoreit. "Scaling autoregressive video models". In: *arXiv preprint arXiv:1906.02634* (2019).

[149] Xinshuo Weng et al. "GNN3DMOT: Graph neural network for 3D multi-object tracking with 2D-3D multi-feature learning". In: *CVPR*. 2020.

[150] Chao-Yuan Wu and Philipp Krahenbuhl. "Towards long-form video understanding". In: *CVPR*. 2021.

[151] Yuxin Wu et al. *Detectron2*. 2019.

[152] Zhirong Wu et al. "Unsupervised feature learning via non-parametric instance discrimination". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 3733–3742.

[153] Bin Xiao, Haiping Wu, and Yichen Wei. "Simple baselines for human pose estimation and tracking". In: *ECCV*. 2018.

[154] Tete Xiao et al. "Masked visual pre-training for motor control". In: (2022).

[155] Yuliang Xiu et al. "Pose Flow: Efficient online pose tracking". In: *BMVC*. 2018.

[156] Jiarui Xu et al. "Spatial-temporal relation networks for multi-object tracking". In: *ICCV*. 2019.

[157] Yihong Xu et al. "How to train your deep multi-object tracker". In: *CVPR*. 2020.

[158] An Yan et al. "PA3D: Pose-action 3D machine for video recognition". In: *CVPR*. 2019.

[159] Greg Yang et al. *Tensor Programs V: Tuning Large Neural Networks via Zero-Shot Hyperparameter Transfer*. 2022. arXiv: `2203.03466 [cs.LG]`.

[160] Alper Yilmaz, Omar Javed, and Mubarak Shah. "Object tracking: A survey". In: *Acm computing surveys (CSUR)* 38.4 (2006), 13–es.

[161] Biao Zhang and Rico Sennrich. "Root mean square layer normalization". In: *Advances in Neural Information Processing Systems* 32 (2019).

[162] Hongwen Zhang et al. "PyMAF: 3D human pose and shape regression with pyramidal mesh alignment feedback loop". In: *ICCV*. 2021.

[163] Jason Y Zhang et al. "Predicting 3D human dynamics from video". In: *ICCV*. 2019.

[164] Shiwen Zhang. “Tfcnet: Temporal fully connected networks for static unbiased temporal reasoning”. In: *arXiv preprint arXiv:2203.05928* (2022).

[165] Yang Zhang et al. “Long-term tracking with deep tracklet association”. In: *TIP* (2020).

[166] Yubo Zhang et al. “A structured model for action detection”. In: *CVPR*. 2019.

[167] Yuxiang Zhang et al. “4D association graph for realtime multi-person motion capture using multiple video cameras”. In: *CVPR*. 2020.

[168] Lianghui Zhu et al. “Vision mamba: Efficient visual representation learning with bidirectional state space model”. In: *arXiv preprint arXiv:2401.09417* (2024).