Beyond the Camera: Neural Networks in World Coordinates

Gunnar A. Sigurdsson [1⋆]
Abhinav Gupta [1] Cordelia Schmid [2] Karteek Alahari [2]
[1]Carnegie Mellon University [2]Inria⋆⋆
github.com/gsig/world-features

Abstract. Eye movement and strategic placement of the visual field onto the retina, gives animals increased resolution of the scene and suppresses distracting information. This fundamental system has been missing from video understanding with deep networks, typically limited to 224 by 224 pixel content locked to the camera frame. We propose a simple idea, WorldFeatures, where each feature at every layer has a spatial transformation, and the feature map is only transformed as needed. We show that a network built with these WorldFeatures, can be used to model eye movements, such as saccades, fixation, and smooth pursuit, even in a batch setting on pre-recorded video. That is, the network can for example use all 224 by 224 pixels to look at a small detail one moment, and the whole scene the next. We show that typical building blocks, such as convolutions and pooling, can be adapted to support WorldFeatures using available tools. Experiments are presented on the Charades, Olympic Sports, and Caltech-UCSD Birds-200-2011 datasets, exploring action recognition, fine-grained recognition, and video stabilization.

1 Introduction

The success of recent vision systems is in a way surprising, since the input is typically a 224×224 pixel image, or in the case of videos, a temporal stack of such images [1,6]. This is both low resolution, and does not give much flexibility to investigate important signals in the image. The system is constrained to what the camera recorded, and has typically no active role in collecting the data. In particular, video recognition architectures have been shown to be vulnerable to camera motion, subject size, and temporal scale [28]. In comparison, the human visual system is not a passive receiver—our eyes are constantly moving to increase the effective resolution of the scene and suppress irrelevant signals [8].

These eye movements exist in humans and animals with foveal vision, and can be categorized as: *Stabilization*, the vestibulo-ocular reflex describes a control system the human visual system uses to stabilize images on the retina given proprioceptive information, like head rotation [3]. *Smooth Pursuit*, where the

⋆ Work was done while Gunnar was at Inria.
⋆⋆ Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France.

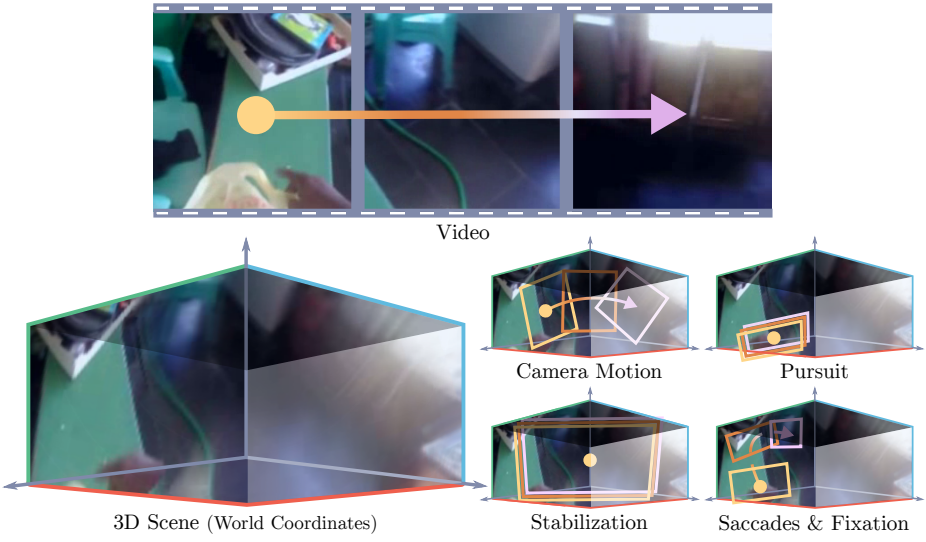

Fig. 1. In this paper we explore how to model the variety of eye motions, even on pre-recorded data. We generalize the concept of feature maps, to WorldFeatures, that include a relative location in space and time, which can encode any type of "eye motion" we want apply to the recorded data, or undo. Orange to pink is used to denote features at time $t-1$, t, and $t+1$. We show a video from a first-person view, along with the *Camera Motion* in the reconstructed world frame. The original video follows a particular sequence of viewpoints, however, crucial information about the activities in the video requires different views—*Stabilization* transforms the views to the world frame, *Pursuit* transforms the views to follow the subject of the video, and *Saccades & Fixation* scans the scene to extract fine-grained information from the video.

gaze is voluntarily shifted to track a moving object, effectively stabilizing the object of interest on the retina [15]. *Fixation*, where gaze is fixed towards a single location, to enhance resolution of that area [24]. *Saccades*, where the eyes quickly jerk between phases of fixation to explore the scene [14]. In Fig. 1 we illustrate how a video is constrained to what the camera recorded, but that also the video may be refocused. How can we build a vision system that has this flexibility to freely explore the data, and move beyond a fixed 224×224 window?

We propose a simple and effective idea—each feature, has a location in real-world coordinates. This combination of `(feature, transformation)` pairs is referred to as a WorldFeature. An example transformation might be image coordinates to real-world coordinates (camera matrix). It turns out preprocessing the video data to, for example, stabilize it, introduces new problems, as demonstrated in Fig. 2. Instead, we use these WorldFeatures in all layers in the network and index the data according to the transformation or transform it as needed. That is, *implicit* transformation instead of *explicit*. With such network, we can utilize any type of "eye motion" we want apply to the recorded data, or undo.

Fig. 2. Illustration of WorldFeatures, attaching transformations to features. Top left is the original video shown as two frame featuremap. We show the temporal average of an aligned feature map, with three di erent transformations T_1 (identity), T_2 (pursuit), T_3 (stabilization). Observe how di erent transforms highlight di erent aspects of the data. We show a potential coordinate frame, that could be used to t the data into a network, highlighting a problem with naively stabilizing the data, for example, when 64 frames in a row need to be aligned. WorldFeatures get around this by keeping the transformation and only using it as needed.

To illustrate the idea, in Fig. 2 we present a simple two frame featuremap with three di erent transformations T_1, T_2, T_3 to showcase how di erent transformations can highlight di erent signals in the data. For example, in the third it is easy to understand the global layout of the scene, and in the second it is easier to understand the di erence in the pose of the person. In Sec. 3 we show more details on WorldFeatures, and how to keep track of the transformations after every layer to build networks with these operations.

Contributions. In this paper we propose the idea of WorldFeatures, where each feature has a transformation, on multiple vision tasks: Fine-grained Image Recognition (Sec. 4.1), Video Stabilization (Sec. 4.2), and Video Activity Recognition (Sec. 4.3). We formalize WorldFeatures, and show how di erent eye movements can be modeled in this framework. Further, we propose a simple implementation that can adapt optimized neural networks tools to operate on WorldFeatures. This implementation can extend many state-of-the-art building blocks that operate over space and time, such as 3D Convolutions, and MaxPooling. The output of each building block, is also WorldFeatures, such that it can be processed hierarchically (e.g. 3D-ResNet [36]).

2 Related Work

Many image and video understanding architectures have been proposed in the last few decades and we refer the reader to [20,37] for a detailed survey, and focus here on recent video architectures related to our approach.

Video Architectures. Recently, networks have become more evolved for separating content and motion [1,25,30,6]. This direction is analogous to the different specialization of the ventral and dorsal streams in the mammalian brain. Similarly, separating camera motion and subject motion has shown great promise in video segmentation [31]. As the field moves towards utilizing longer temporal context [38], equipping these networks with the capability to separate camera motion, object permanence, scene dynamics, and subject motion can greatly improve those efforts. This paper aims to provide the building blocks necessary to construct these systems.

Video Stabilization. Related to our work on understanding eye motion, is the task of video stabilization that has been pursued directly [9,35,18], or with additional supervision [17]. Of particular relevance is recent work that has incorporated stabilization and tracking of points over time to improve accuracy of video recognition [33,34]. Inspired by these successes, we go a step further, and build a framework that can operate given an arbitrary transformation signal, and use this to improve the vision system.

Improving Visual Input. On the other hand, recent work has also explored how to use the input data more effectively. This has been done through spatially or temporally modifiable connections [4], non-local connections where feature similarity guides the connections [36], or transformations of the input [13]. Furthermore, [22] investigated the idea that not all input features are equally useful, and explored how to learn how to highlight and zoom in on the important content in each image. As we will see, eye movement is primarily useful to allow for efficient allocation of resources, and since our system can handle arbitrary transformations, we can explicitly utilize various zoom in of the data. Furthermore, our WorldFeatures is a system that allows a network to generalize beyond the camera, and is complementary to frameworks that learn attention to emphasize parts of the data.

3 WorldFeatures: Feature Maps with Camera Movements

We start by introducing the ideas behind WorldFeatures, that is, (feature, transformation) pairs. Then we demonstrate how different eye movements can be modeled in this framework. A video is a specific window into the world at the time it was recorded, and a system analyzing this video might have a completely different objective for watching the scene than the camera operator. To undo the camera bias or highlight signals in the video, we allow a transformations that are analogous to eye movements, after the data has already been recorded.

To undo camera transformations, could we pre-process the data? For example, applying affine transformations to each frame to undo camera movement, for stabilization. In fact, even this simple case has problems: How to fit this transformed data into the model? At what scale should we process the data? How to use imperfect transforms? Consider eyes scanning a scene, fixating on a variety of points in the scene at multiple resolutions. Creating data that follows

Fig. 3. Demonstration of a model using WorldFeatures at each layer. Here we look at an illustration of the outputs of conv1 and maxpool layers, when applied to an eye movement task. The input and output of each layer are WorldFeatures. The receptive field of each output featuremap is aligned, explicitly or implicitly depending on implementation, before computing the output. Since each output feature map has a different alignment, each output has a different transformation, and we repeat the process at every layer. This shows that WorldFeatures allow the network to combine information across multiple locations and scales in the image.

a similar pattern without an understanding how they relate to each other creates a noisy and disjointed view of the scene. Instead, by keeping track of the relative transformations between features, we can crop, scale, and zoom in on the data as required, while still maintaining correct relationships between features.

3.1 WorldFeatures Definition

The key to our idea is the concept of WorldFeatures x^T. That is, a feature map x of size $T \times H \times W$ (time, height, width) and an arbitrary transform T which can encode any type of eye motion we want to apply to the recorded data or undo. [?] We refer to a single $H \times W$ timepoint as a frame, and note that $x = x^I$, where I is the identity transform. The feature of x^T at $(t;h;w)$ is written $x(T(t;h;w))$. Therefore, extending a convolution operation to work on x^T:

$$y^{T'}(t;h;w) = \sum_i \sum_j \sum_k x(T(i;j;k)) \; f(t-i;h-j;w-k); \qquad (1)$$

where $y^{T'}$ is also a WorldFeature, that can then be passed to the next convolution operation and so on. Here f is a filter, and we omit batch and channel dimensions

[?] For example, the affine transform, $T(t;h;w) = (t;\ a_t^0 h + a_t^1 w + a_t^2;\ a_t^3 h + a_t^4 w + a_t^5)$, where a_t^n are the parameters at frame t.