# Motion Graph Unleashed:
# A Novel Approach to Video Prediction

**Yiqi Zhong**[13*]    **Luming Liang**[1*]    **Bohan Tang**[2]    **Ilya Zharkov**[1]    **Ulrich Neumann**[3]

[1]Microsoft    [2]University of Oxford    [3]University of Southern California

[1]{yiqizhong,lulian,zharkov}@microsoft.com

[2]bohan.tang@eng.ox.ac.uk    [3]{yiqizhon,uneumann}@usc.edu

## Abstract

We introduce *motion graph*, a novel approach to the video prediction problem, which predicts future video frames from limited past data. The motion graph transforms patches of video frames into interconnected graph nodes, to comprehensively describe the spatial-temporal relationships among them. This representation overcomes the limitations of existing motion representations such as image differences, optical flow, and motion matrix that either fall short in capturing complex motion patterns or suffer from excessive memory consumption. We further present a video prediction pipeline empowered by motion graph, exhibiting substantial performance improvements and cost reductions. Experiments on various datasets, including UCF Sports, KITTI and Cityscapes, highlight the strong representative ability of motion graph. Especially on UCF Sports, our method matches and outperforms the SOTA methods with a significant reduction in model size by $78\%$ and a substantial decrease in GPU memory utilization by $47\%$. Please refer to this link for the official code.

## 1   Introduction

Video prediction aims at predicting future frames given a limited number of past frames. This technology has potential for numerous applications, including video compression, visual robotics, and surveillance systems. A critical aspect of designing an effective video prediction system is the precise modeling of motion information, which is essential for its successful deployment.
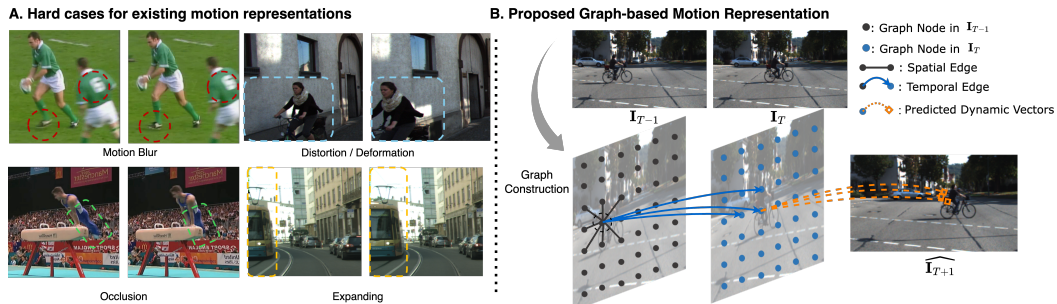


Figure 1: (A) Hard cases which cannot be properly modeled by most existing representations. (B) Motion graph transforms single-frame patches into interconnected nodes, describing the spatial-temporal relationships. Future per-pixel motion dynamic vectors are then predicted on this graph.

---

[1]Equal contributions. The work is mostly done during Yiqi Zhong's internship at Microsoft.

https://doi.org/10.52202/079017-3525

At first glance, video prediction appears as a sequence prediction problem, with conventional approaches using advanced sequential data modeling techniques such as 3D convolution [1, 2], recurrent neural networks [3, 4, 5, 6, 7], and transformers [8, 9, 10]. These methods implicitly model motion and image appearance to help reason sequence evolution. However, the implicit approach to motion modeling can result in inefficient learning, which stems from the inherent difference between video prediction and typical sequence prediction: in typical sequence prediction, features tend to evolve uniformly; whereas video prediction involves not only static elements like object appearance but also dynamic elements intricately embedded in the sequence, e.g., motion-related information, including object pose and camera movement. Effectively managing both the static and dynamic aspects of the hidden feature space presents a great challenge.

Instead of treating video prediction purely as a sequence prediction problem, an alternative approach is to explicitly model inter-frame motion, and redefine the task primarily as a motion prediction problem [11, 12, 13, 10, 2]. For this approach, it is crucial to select an appropriate motion representation, which should exhibit expressiveness by accurately encapsulating motion with detailed information like direction and speed. However, as Section 2 and Table 1 will elaborate, existing video motion representations have limitations. Some, as Figure 1(A) demonstrates, lack representative ability for complex scenarios. For example, optical flow is unable to handle motion blur and object distortion/deformation resulting from perspective projection. Others over-sacrifice memory efficiency to model complex motion patterns, such as the motion matrices proposed in MMVP [2].

To fill the gap in existing research, we propose a novel motion representation: the motion graph, which treats image patches of the observed video frames in the downsample space as interconnected graph nodes based on their spatial and temporal proximity; See Figure 1 (B) for an illustration. This approach achieves both goals of representativeness and compactness by generating accurate predictions while conserving memory and computational resources. For the representative ability, our motion graph enriches each node with dynamic information, including multiple feasible weighted flows toward the subsequent frame. This method captures a broader range of motion patterns compared to single-direction representations like optical flow and enhances error tolerance. Additionally, the motion graph excels in compactness by avoiding computation-heavy operations such as stacked convolutional layers. To reduce computational complexity, we also limit the connections in the motion graph to a fixed number of neighbors for each node. Consequently, compared to previous representations (e.g., motion matrix, keypoint heatmap), the motion graph offers a more sparse and memory-efficient modeling of motion, effectively transcending the limitations of 2D image structures.

Based on the motion graph we have introduced, we present a novel video prediction pipeline. The representative ability and compactness of the motion graph allow our system to have achieved notable enhancements in computational efficiency. We test various scenarios with different motion patterns on three well-known datasets, UCF Sports, KITTI, and Cityscapes. Across all three datasets, our approach consistently exhibits performance that aligns with or surpasses the state-of-the-art (SOTA) methods with significantly lower GPU comsumption.

## 2   Related Works

In this section, we mainly discuss the existing video prediction systems that apply to the setting of short-term high resolution video prediction, please see more details about the setting in Section A.5.

Based on whether they explicitly model motion, existing video prediction methods can be put into two categories. Methods that do not explicitly model motion usually consider video prediction purely as a sequence prediction problem [3]. They use advanced sequential modeling techniques such as recurrent neural networks [4, 3, 5, 14, 15, 16, 17, 18] and transformers [8, 10, 19, 20] to estimate the evolution of image features in the hidden space along the temporal dimension. However, video prediction holds a unique property that is unlike text generation or other sequence prediction problems: that is, most features in video sequences (e.g., object appearance, scene layout, texture of the background) usually remain unchanged, or have no major swift change, from frame to frame. Yet, sequential modeling techniques treat the whole image features uniformly; using them for video prediction thus requires modification in the inner structure of certain sequential modeling architectures to accommodate this property of video prediction. Some works choose to enhance the spatial consistency by adding complex residual shortcuts [4, 17, 21], or by decomposing temporal and spatial information into two separate hidden spaces, hoping that the network will figure out which to maintain and which

to change [15, 16]. Although such accommodations sometimes help systems reach impressive prediction accuracy (given the powerful sequential modeling capability of the advanced techniques), these methods tend to have larger model sizes and complicated architectures.

To improve the efficiency of motion modeling in systems, this paper extends the idea shared by the second type of methods, which treat video prediction as a motion prediction problem, and explicitly model the motion hidden in image sequences. Such methods either reason future motion patterns using certain representations with image features as the input, or solely use motion representations as the input for prediction. There are four lines of work on motion representations. First, image difference [11, 10] uses the subtraction of two consecutive frames to represent the dynamic of image sequences. It requires almost no computational cost to calculate but does not directly describe motion. Second, optical flow/ voxel flow [22, 23, 24, 13] describes the pixel-level motion. But, it usually requires auxiliary or out-of-shell models to generate, and is limited to only modeling one-to-one relationships [2], which can be a disadvantage, especially for scenarios with ambiguous pixel-to-pixel correspondence (e.g., motion blur). Third, key point trace [12] serves highly structured scenarios (e.g., videos with static backgrounds and dominated by human motions) but can be inefficient for more complex content, including crowded scenes with multiple objects. Fourth, the recent motion matrix [2] describes the all-pair relationship between consecutive frames to overcome the drawback of optical flow. While motion matrix supports a more efficient and compact system of a notably smaller size, concerns were over the computational and space complexity of its operations [2].

Table 1: Motion representation comparison. We assess the representative ability by judging how accurate the motion can be described, especially for the hard cases demonstrated in Fig 1(A).

| Motion Representation | Out-of-shell Model | Representative Ability | Space Complexity |
| --- | --- | --- | --- |
| Image difference [11, 10] | n/a | low | $O(n)$ |
| Keypoint trace [12] | required | medium | $O(n)$ |
| Optical flow/ voxel flow [22, 23, 24, 13] | Some required | medium | $O(n)$ |
| Motion matrix [2] | n/a | high | $O(n^2)$ |
| Motion graph (ours) | n/a | high | $O(n)$ |

Besides the above motion representations, we notice that graph, as a sparse data structure that can accurately describe the relationships between entities, has been widely adopted in ordinary motion prediction systems, such as trajectory forecasting [25, 26, 27] and human motion prediction [28, 29, 30]. Inspired by these successful adoptions, we propose to represent the motion in videos using graph structures, named *motion graph* in this work.

The major difference between motion graph and previous motion representations is that motion graph can describe many-to-many temporal and spatial correspondence with low space complexity. Such sparse representations not only enhance representative ability but also advantageously i) replace most 2D convolution operations by graph operations which are mostly implemented by linear layers, leading to faster and more parameter-efficient systems; ii) enable convenient spatial and temporal interactions among feature patches, promoting higher learning efficiency and prediction accuracy.

## 3 Methodology

In this work, video prediction is approached as a motion prediction problem, utilizing a novel representation called the motion graph. This representation is designed to effectively capture the intricate motion dynamics within video sequences. Leveraging the motion graph, we develop a video prediction pipeline that achieves high performance with reduced model size and GPU memory requirements. Section 3.1 details our problem formulation and notation. Section 3.2 explains the construction of the motion graph from input video sequences. Finally, Section 3.3 delves into our video prediction pipeline, illustrating how the motion graph is used for effective video prediction.

### 3.1 Problem Formulation

Given a video sequence with $T$ frames $\{\mathbf{I}_t \in \mathbb{R}^{H \times W \times 3} | t = 0, 1, ..., T - 1\}$, a video prediction system aims to predict the next $T'$ frames $\{\mathbf{I}_{t'} \in \mathbb{R}^{H \times W \times 3} | t' = T, T + 1, ..., T + T' - 1\}$. We approach this task by regarding video prediction as a pixel-level motion prediction problem.
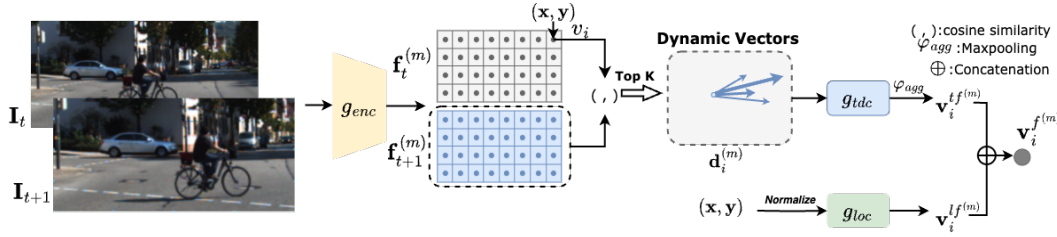
Figure 2: **Motion graph node construction**: Cosine similarity, denoted by $(,)$, between patch features in consecutive frames is computed to further choose top $k$ directions for each patch. Tendency $\mathbf{v}_i^{tf^{(m)}}$ and location features $\mathbf{v}_i^{lf^{(m)}}$ are then generated based on these $k$ vectors and the patch location.

Consider a pixel located at $(x, y)$ in a given video frame $\mathbf{I}_t$. Our system is designed to predict $k$ dynamic vectors starting from this pixel, pointing to the possible locations in the target future frame $\mathbf{I}_{t'}$. These vectors represent the pixel's anticipated motion from its current position in $\mathbf{I}_t$ to its future position in $\mathbf{I}_{t'}$. Mathematically, we express the dynamic vectors of all pixels across the observed frames as $\mathbf{P} \in \mathbb{R}^{T \times H \times W \times k \times 3}$, where $\mathbf{P}_{t,x,y} = \left[ [\Delta x_1, \Delta y_1, w_1], ..., [\Delta x_k, \Delta y_k, w_k] \right]$ with both the motion direction $(\Delta x_k, \Delta y_k)$ and the weighted component $(w_k)$. The transformation of previous frames into future frames is performed by a pixel-level image warper, denoted as $\overrightarrow{\mathcal{W}}$. The warping is based on the predicted dynamic vectors and is formally defined by the following equation:

$$\widehat{\mathbf{I}_T} = \overrightarrow{\mathcal{W}}(\mathbf{P}, \mathbf{I}_0, ..., \mathbf{I}_{T-1}). \tag{1}$$

This formulation allows us to capture and translate the intricate motion dynamics within the video sequence into future frame predictions with enhanced accuracy.

### 3.2 Motion Graph

#### 3.2.1 Intuition

Inspired by advancements in general motion prediction systems [25, 26, 30], we posit that modeling semantic correlations among image patches within observed frames is essential for improving current motion representations in video prediction. To capture such correlations, we first model the semantic information of each image patch across a video with $T$ observed frames. This is achieved by multi-scale image patch feature mappings, $\mathcal{F} = \{\mathbf{f}^{(1)}, \cdots, \mathbf{f}^{(M)}\}$, generated by inputting each observed frame into a shared image encoder $g_{enc}$ and applying a pixel unshuffle technique [31] to reshape all $M$ feature maps to the resolution of the smallest features. Here, the smallest resolution of the features generated by $g_{enc}$ is $H_s \times W_s$, and $\mathbf{f}^{(m)} \in \mathbb{R}^{T \times H_s \times W_s \times C_m}$ represents the feature map in the $m$-th scale containing the $C_m$-dimensional features of $T \times H_s \times W_s$ distinct image patches.

Based on $\mathcal{F}$, we construct a multi-view graph, named *motion graph*, to capture the semantic correlations among $T \times H_s \times W_s$ image patches. This graph is represented as: $\mathcal{G} = \{\mathcal{V}, \mathcal{E}^{(1)}, \cdots, \mathcal{E}^{(M)}\}$, where $\mathcal{V} = \{v_0, \cdots, v_{TH_sW_s-1}\}$ denotes the set of nodes each corresponding to an image patch, and $\mathcal{E}^{(m)}$ denotes the edge set for the $m$-th view capturing the correlations driven by the semantic information within $\mathbf{f}^{(m)}$. We detail the graph construction and the graph-related operations as follows.

#### 3.2.2 Node Motion Feature Initialization

For each node $v_i$ in the $m$-th view, we initialize its motion feature $\mathbf{v}_i^{f^{(m)}} \in \mathbb{R}^{d_{mf}^{(m)}}$ with two components: the tendency feature $\mathbf{v}_i^{tf^{(m)}} \in \mathbb{R}^{d_{tf}^{(m)}}$ and the location feature $\mathbf{v}_i^{lf^{(m)}} \in \mathbb{R}^{d_{lf}^{(m)}}$. The tendency feature $\mathbf{v}_i^{tf^{(m)}}$ captures the node's motion-related attributes relative to nodes in the subsequent frame and the location feature $\mathbf{v}_i^{lf^{(m)}}$ models the normalized absolute location of each node in a frame.

Inspired by prior works [2, 32], we generate the tendency feature $\mathbf{v}_i^{tf^{(m)}}$ in three steps. First, for each node $v_i$, we form a node list $\mathcal{L}_i^{(m)}$ by choosing top-$K$ scoring image patches from the subsequent frame based on the cosine similarities computed by $\mathbf{f}^{(m)}_{\lfloor \frac{i}{H_sW_s} \rfloor}$ and $\mathbf{f}^{(m)}_{\lfloor \frac{i}{H_sW_s} \rfloor+1}$. By doing so, we mitigate

the risk of false positives and better accommodate complex motion patterns, especially in downscaled spaces where an image patch may exhibit multiple potential movements. Second, we define the dynamic vector for each node $v_i$ as $\mathbf{d}_i^{(m)} = [\Delta x_1, \Delta y_1, w_1, \cdots, \Delta x_K, \Delta y_K, w_K] \in \mathbb{R}^{3K}$, where $\Delta x_k, \Delta y_k$ indicate the motion direction and $w_k$ is the cosine similarity score between $v_i$ and the $k$-th node in $\mathcal{L}_i^{(m)}$. Here, an exception is made for nodes in the last observed frame $\mathbf{I}_{T-1}$, where we apply zero-padding to the dynamic vectors, as $\mathbf{I}_T$ is unknown. Finally, we generate the tendency feature $\mathbf{v}_i^{tf^{(m)}}$ for each node by feeding each dynamic vector $\mathbf{d}_i^{(m)}$ into a multilayer perceptron (MLP) $g_{tdc}(.)$ followed by a max-pooling operation $\varphi_{agg}(.)$, as shown in:

$$\mathbf{v}_i^{tf^{(m)}} = \varphi_{agg}(g_{tdc}(\mathbf{d}_i^{(m)})). \tag{2}$$

The location feature $\mathbf{v}_i^{lf^{(m)}}$ encodes the position of a pixel in a frame. Pixel positions influence motion patterns. For instance, pixels on the sides of a frame may appear to move differently than pixels in the center for street views collected by wide-range moving cameras, due to the perspective projection effect. We use another MLP $g_{loc}(.)$ to extract the location feature and define it as:

$$\mathbf{v}_i^{lf^{(m)}} = g_{loc}\left(\frac{x}{W_s}, \frac{y}{H_s}\right), \tag{3}$$

Finally, the motion features of each node $\mathbf{v}_i^{f^{(m)}}$ are initialized as follows:

$$\mathbf{v}_i^{f^{(m)}} = \oplus(\mathbf{v}_i^{lf^{(m)}}, \mathbf{v}_i^{tf^{(m)}}), \tag{4}$$

where $\oplus(\cdot)$ denotes the concatenation operation. Figure 2 details the node motion feature initialisation.

### 3.2.3 Edge Construction

After initializing the node features, we construct edges in the motion graph to capture the semantic relationships between image patches in the observed frames. We further represent the edge set of the $m$-th view as $\mathcal{E}^{(m)} = \{\mathcal{E}^{S^{(m)}}, \mathcal{E}^{B^{(m)}}, \mathcal{E}^{F^{(m)}}\}$, where $\mathcal{E}^{S^{(m)}}$ is spatial edges, $\mathcal{E}^{B^{(m)}}$ is backward edges, and $\mathcal{E}^{F^{(m)}}$ is forward edges. Generally, spatial edges are posit on that neighboring image patches in a frame likely influence each other's future motion, and backward and forward edges connect nodes across adjacent frames indicating potential motion paths. Notably, nodes in the first frame are not assigned backward edges, and nodes in the last frame are not assigned forward edges.

We construct these edges in two steps: 1) finding the neighbors of each node connected by spatial, backward, and forward edges respectively; and 2) generating the three types of edges by connecting the neighboring nodes found. For the first step, we denote $\mathcal{N}_i^{S^{(m)}}$, $\mathcal{N}_i^{B^{(m)}}$, and $\mathcal{N}_i^{F^{(m)}}$ as sets containing neighbors of $v_i$ connected by the spatial, backward and forward edges, respectively. Then, the neighbors of each node $v_i$ are found by solving the following optimization problems:

$$\mathcal{N}_i^{S^{(m)}} = \arg\max_{\Omega} ||\mathbf{C}_{i,\Omega}^{(m)}||_{1,1} \ \ s.t. \ \ \Omega \subseteq \mathcal{S}_i, \ \ |\Omega| = k,$$

$$\mathcal{N}_i^{B^{(m)}} = \arg\max_{\Omega} ||\mathbf{C}_{i,\Omega}^{(m)}||_{1,1} \ \ s.t. \ \ \Omega \subseteq \mathcal{B}_i, \ \ |\Omega| = k, \tag{5}$$

$$\mathcal{N}_i^{F^{(m)}} = \arg\max_{\Omega} ||\mathbf{C}_{i,\Omega}^{(m)}||_{1,1} \ \ s.t. \ \ \Omega \subseteq \mathcal{F}_i, \ \ |\Omega| = k,$$

where $\mathbf{C}^{(m)} \in \mathbb{R}^{TH_sW_s \times TH_sW_s}$ contains cosine similarity scores between nodes in $\mathcal{V}$ computed by $\mathbf{f}^{(m)}$, $k \in \mathbb{Z}$ is a hyperparameter, $\mathcal{S}_i = \{v_j \in \mathcal{V} : \lfloor \frac{i}{H_sW_s} \rfloor = \lfloor \frac{j}{H_sW_s} \rfloor\}$, $\mathcal{B}_i = \{v_j \in \mathcal{V} : \lfloor \frac{i}{H_sW_s} \rfloor = \lfloor \frac{j}{H_sW_s} \rfloor + 1\}$, and $\mathcal{F}_i = \{v_j \in \mathcal{V} : \lfloor \frac{i}{H_sW_s} \rfloor = \lfloor \frac{j}{H_sW_s} \rfloor - 1\}$. For the second step, we construct $\mathcal{E}^{S^{(m)}}$, $\mathcal{E}^{B^{(m)}}$, and $\mathcal{E}^{F^{(m)}}$ by connecting each $v_i$ with nodes in $\mathcal{N}_i^{S^{(m)}}$, $\mathcal{N}_i^{B^{(m)}}$, and $\mathcal{N}_i^{F^{(m)}}$ respectively.

### 3.2.4 Motion Graph Interaction Module

After constructing the nodes and establishing the edges in the motion graph, we enable information flow within the $m$-th view of the graph via the message-passing operation $g_{mp}$ defined as follows:

$$\mathbf{v}'^{(m)} = g_{mp}\left(\mathbf{v}^{(m)}, \mathcal{E}^{in^{(m)}}\right), \tag{6}$$

$$\text{Spatial} : \mathbf{v}'^{(m)} = g_{mp}(\mathbf{v}^{(m)}, \mathcal{E}^{S^{(m)}})$$

$$\text{Temporal Forward} : \mathbf{v}'^{(m)} = g_{mp}(\mathbf{v}^{(m)}, \mathcal{E}^{F^{(m)}})$$

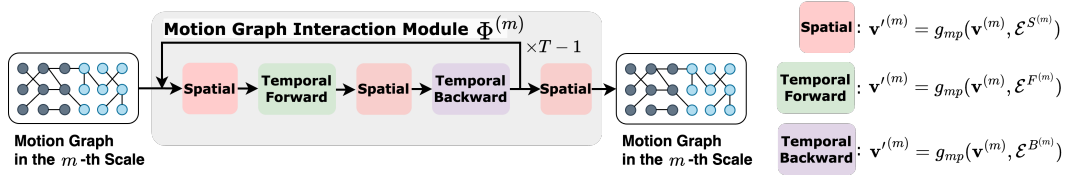$$\text{Temporal Backward} : \mathbf{v}'^{(m)} = g_{mp}(\mathbf{v}^{(m)}, \mathcal{E}^{B^{(m)}})$$

Figure 3: **Inside the interaction module for the** $m$**-th view** $\Phi^{(m)}$. The spatial and temporal message passing are iteratively conducted and repeated $T - 1$ times, where $T$ is the observed frame number.

where $\mathbf{v}^{(m)} \in \mathbb{R}^{TH_sW_s \times d^{(m)}}$ is the input node features, $\mathbf{v}'^{(m)} \in \mathbb{R}^{TH_sW_s \times d^{(m)}}$ is the updated node features, and $\mathcal{E}^{in^{(m)}}$ denotes an edge set. The operation $g_{mp}$ facilitates the information exchange between nodes connected by edges in $\mathcal{E}^{in^{(m)}}$. In practice, the spatial message passing is implemented as a 2D convolution layer, while for the temporal message passing we use graph neural network to implement $g_{mp}$. See implementation details in the appendix.

With the message-passing operation $g_{mp}$, we introduce a motion graph interaction module, denoted as $\Phi$ and illustrated in Figure 3. The goal of $\Phi$ is to ensure that even nodes in the last observed frame are informed about the motion dynamics from the first observed frame, and vice versa. To achieve this comprehensive information flow, we implement $T-1$ rounds of full information transition, where $T$ is the total number of observed frames. Each round of transition includes twice the spatial message passing, once the temporal forward message passing, and once the temporal backward message passing. The combination of spatial and temporal interaction in $\Phi$ ensures holistic information integration for accurate and comprehensive motion prediction in video sequences. The design of $\Phi$ allows for thorough and balanced dissemination of motion information throughout the motion graph.

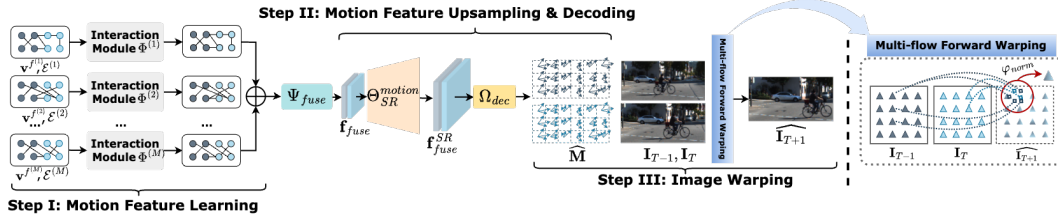### 3.3 Motion-graph-empowered Video Prediction



Figure 4: **Pipeline overview**. After decoding per-pixel motion features into dynamic vectors, we perform multi-flow forward warping for future frame generation.

By constructing the motion graph, we create a tool to extract motion information from the observed video frames. The following passage describes how to conduct video prediction using the constructed motion graph and its related operations. As Figure 4 indicates, to predict the unknown video frame involves three main steps, which are elaborated in the following three subsections.

#### 3.3.1 Motion Feature Learning

**Multi-view motion feature update.** After having the motion graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}^{(1)}, \cdots, \mathcal{E}^{(M)}\}$, with the node motion feature $\mathbf{v}^{f^{(m)}}$ initialized by Eq. (4) in the $m$-th view, we use the motion graph interaction module introduced in Section 3.2.4 to update $\mathbf{v}^{f^{(m)}}$ via the following formula:

$$\hat{\mathbf{v}}^{f^{(m)}} = \Phi^{(m)}\left(\mathbf{v}^{f^{(m)}}, \mathcal{E}^{(m)}\right), \tag{7}$$

where $\hat{\mathbf{v}}^{f^{(m)}} \in \mathbb{R}^{TH_sW_s \times d_{mf}^{(m)}}$ is the updated node features in the $m$-th view of the motion graph.

**Multi-view motion feature fusion.** Once the node feature in each view of the motion graph is updated, we concatenate the node features from each view and apply a fusion module $\Psi_{fuse}$ to integrate these multi-view features into a unified representation as follows:

$$\mathbf{f}_{fuse} = \Psi_{fuse}(\varphi_{cat}(\mathbf{v}^{f^{(m)}}|m = 1, ..., M)), \tag{8}$$

where $\mathbf{f}_{fuse} \in \mathbb{R}^{T \times H_s \times W_s \times C_{node}}$, with $C_{node}$ denoting the dimension of the node features.

### 3.3.2 Motion Feature Upsampling and Decoding

In this step, we transform the fused multi-view motion feature $\mathbf{f}_{fuse}$ into a 2D structure with a resolution of $H_s \times W_s$. This feature is then upscaled to match the resolution of the original video frames ($H \times W$) using a motion upsampler, $\Theta_{SR}$. The architecture of $\Theta_{SR}$ is inspired by ResNet-based image super-resolution networks [33], which progressively refine features from lower to higher resolutions until reaching the original image size. This upsampling process is key to predicting pixel-level motion features for all observed frames, which extend toward the next future frame. By incrementally adjusting the resolution, $\Theta_{SR}$ effectively bridges the gap between the multi-view motion information and the high-resolution requirements of accurate pixel-level motion prediction.

Upon obtaining the pixel-level motion feature $\mathbf{f}_{fuse}^{SR}$, we use a motion decoder, $\Omega_{dec}$, to convert it into pixel-level dynamic vectors $\mathbf{P}$. As defined in Section 3.1, $\mathbf{P}$ contains $k$ potential motion directions with associated probability scores for each pixel. This decoding output is compatible with the motion features in the motion graph, which are generated by the $k$ dynamic vectors of image patches.

### 3.3.3 Image Warping

After having the dynamic vectors through the decoding process, we use them to warp the observed frames into the future frame. Unlike traditional methods that often use optical-flow-based backward warping, our approach follows the design logic of the motion graph feature learning as well as the output format to use a multi-flow forward warping technique. This method, drawing from the image splatting concepts in prior works [34, 35], is visually detailed in Figure 4. Forward warping allows each predicted dynamic vector to directly contribute to the construction of the future frame. To aggregate the contributions of multiple vectors at each pixel location in the synthesized frame, we apply a normalization operation $\varphi_{norm}$. It normalizes the weight of each vector based on the sum of their weights, ensuring an even and balanced contribution to the final pixel value in the future frame.

## 4 Experiments

For evaluation, we trained our video prediction pipeline in an end-to-end fashion on three public datasets: i) **UCF Sports** [36] (i.e. 150 video sequences emphasizing various sports scenes, with frame resolution, and two different data splits to date—one from STRPM [15], one from MMVP [2]— evaluation is on both splits); ii) **KITTI** [37] (i.e. 28 driving videos with a resolution of $375 \times 1242$; following previous works [38, 13], the image frames are resized to $256 \times 832$); and iii) **Cityscapes** [39] (i.e. 3,475 driving videos with 2,945 in the training set and 500 in the validation set). For each dataset, by default, we set $k = max(10, 1\% \times H_s \times W_s)$ which is $1\%$ of the smallest feature map's resolution and no larger than 10 for efficiency consideration. Table 2 shows the configuration for each dastaset. More training and implementation details are in the appendix.

Table 2: Dataset configurations

| Dataset | Resolution | $H_s$ | $W_s$ | Input Frame | Output Frame | Training Loss | k |
|---|---|---|---|---|---|---|---|
| UCF Sports | $512 \times 512$ | 32 | 32 | 4 | 1 | mean-square error (following [15, 16, 2]) | 10 |
| KITTI | $256 \times 832$ | 16 | 52 | 2 | 1 | l1 + Perceptual loss(following [13]) | 8 |
| Cityscapes | $512 \times 1024$ | 32 | 64 | 2 | 1 | l1 + Perceptual loss (following [13]) | 10 |

### 4.1 Public Benchmark Comparison

On the UCF Sports STRPM split, we evaluate the method on two metrics following previous research [15]: peak signal-to-noise ratio (PSNR) and learned perceptual image patch similarity (LPIPS) [40]. The proposed method is compared with existing methods for their results of the 1st and 6th future frames ($t = 5$, $t = 10$ in Table 3). Table 3 shows that our method matches the SOTA performance on the PSNR metric and outperforms all previous methods on the LPIPS metric. On the UCF Sports MMVP split, the validation dataset has been divided into three categories: the easy (SSIM $\leq 0.9$), intermediate ($0.6 \leq$ SSIM $< 0.9$), and hard subsets (SSIM $< 0.6$), which take up $66\%$, $26\%$, and $8\%$ of the full set respectively. We evaluate the methods on PSNR, LPIPS, and Structural Similarity Index Measure (SSIM). Table 4 shows that the proposed method outperforms all other methods.

Table 3: Performance comparison on UCF Sports STRPM split

| Method | $t = 5$ | | $t = 10$ | |
|---|---|---|---|---|
| | PSNR $\uparrow$ | LPIPS$_{\times 100}$ $\downarrow$ | PSNR $\uparrow$ | LPIPS$_{\times 100}$ $\downarrow$ |
| ConvLSTM (NeurIPS 2015) [3] | 26.43 | 32.20 | 17.80 | 58.78 |
| BeyondMSE (ICLR 2016) [41] | 26.42 | 29.01 | 18.46 | 55.28 |
| PredRNN (NeurIPS 2017) [4] | 27.17 | 28.15 | 19.65 | 55.34 |
| PredRNN++ (ICML 2018) [5] | 27.26 | 26.80 | 19.67 | 56.79 |
| SAVP (arXiv 2018) [42] | 27.35 | 25.45 | 19.90 | 49.91 |
| SV2P (ICLR 2018) [43] | 27.44 | 25.89 | 19.97 | 51.33 |
| E3D-LSTM (ICLR 2019) [17] | 27.98 | 25.13 | 20.33 | 47.76 |
| CycleGAN (CVPR 2019) [44] | 27.99 | 22.95 | 19.99 | 44.93 |
| CrevNet (ICLR 2020) [45] | 28.23 | 23.87 | 20.33 | 48.15 |
| MotionRNN (CVPR 2021) [7] | 27.67 | 24.23 | 20.01 | 49.20 |
| STRPM (CVPR 2022) [15] | 28.54 | 20.69 | 20.59 | 41.11 |
| STIP (arXiv 2022) [16] | 30.75 | 12.73 | 21.83 | 39.67 |
| DMVFN (CVPR 2023) [13] | 30.05 | 10.24 | 22.67 | 22.50 |
| MMVP (ICCV 2023) [2] | 31.68 | 7.88 | 23.25 | 22.24 |
| Ours | **31.70** | **6.61** | **23.27** | **19.94** |

Table 4: Performance comparison on the UCF Sports MMVP split.

| Method | Full set | | | Easy (SSIM $\geq$ 0.9) | | | Intermediate (0.6 $\leq$ SSIM < 0.9) | | | Hard (SSIM < 0.6) | | | Model Size$\downarrow$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | SSIM $\uparrow$ | PSNR $\uparrow$ | LPIPS $\downarrow$ | SSIM $\uparrow$ | PSNR $\uparrow$ | LPIPS $\downarrow$ | SSIM $\uparrow$ | PSNR $\uparrow$ | LPIPS $\downarrow$ | SSIM $\uparrow$ | PSNR $\uparrow$ | LPIPS $\downarrow$ | |
| STIP [16] | 0.8817 | 28.17 | 0.1626 | 0.9491 | 30.65 | 0.1066 | 0.8351 | 23.97 | 0.2271 | 0.4673 | 15.97 | 0.4450 | 18.05M |
| SimVP [1] | 0.9189 | 29.97 | 0.1326 | 0.9664 | 32.87 | 0.0584 | 0.8845 | 25.79 | 0.1951 | 0.6267 | 18.99 | 0.5600 | 3.47M |
| MMVP [2] | 0.9300 | 30.35 | 0.1062 | 0.9674 | 33.05 | 0.0580 | 0.8970 | 26.29 | 0.1569 | 0.7203 | **20.84** | 0.3510 | 2.79M |
| Ours | **0.9314** | **30.49** | **0.0823** | **0.9685** | **33.23** | **0.0444** | **0.8978** | **26.36** | **0.1348** | **0.7264** | 20.83 | **0.2320** | **0.60M** |

Table 5: Evaluation on Cityscapes [39] and KITTI [37] datasets. "RGB", "F", "S" and "I" denote video frames, optical flow, semantic map, and instance map; $t + 3$ and $t + 5$ respectively indicate the average performance of the next 3 and 5 frames. Results with * are copied from [13].

| Method | Input | Cityscapes | | | | | | KITTI | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | MS_SSIM($\times 10^{-2}$)$\uparrow$ | | | LPIPS($\times 10^{-2}$)$\downarrow$ | | | MS_SSIM($\times 10^{-2}$)$\uparrow$ | | | LPIPS($\times 10^{-2}$)$\downarrow$ | | |
| | | $t+1$ | $t+3$ | $t+5$ | $t+1$ | $t+3$ | $t+5$ | $t+1$ | $t+3$ | $t+5$ | $t+1$ | $t+3$ | $t+5$ |
| Vid2vid (NeurIPS 2018)* [46] | RGB+S | 88.16 | 80.55 | 75.13 | 10.58 | 15.92 | 20.14 | N/A | N/A | N/A | N/A | N/A | N/A |
| Seg2vid (CVPR 2019)* [47] | RGB+S | 88.32 | N/A | 61.63 | 9.69 | N/A | 25.99 | N/A | N/A | N/A | N/A | N/A | N/A |
| FVS (CVPR 2020)* [48] | RGB+S+I | 89.1 | 81.13 | 75.68 | 8.5 | 12.98 | 16.5 | 79.28 | 67.65 | 60.77 | 18.48 | 24.61 | 30.49 |
| SADM (CVPR 2021)* [49] | RGB+S+F | **95.99** | N/A | 83.51 | 7.67 | N/A | 14.93 | 83.06 | 72.44 | 64.72 | 14.41 | 24.58 | 31.16 |
| PredNet (ICLR 2017)* [50] | RGB | 84.03 | 79.25 | 75.21 | 25.99 | 29.99 | 36.03 | 56.26 | 51.47 | 47.56 | 55.35 | 58.66 | 62.95 |
| MCNET (ICLR 2017)* [11] | RGB | 89.69 | 78.07 | 70.58 | 18.88 | 31.34 | 37.34 | 75.35 | 63.52 | 55.48 | 24.05 | 31.71 | 37.39 |
| DVF (ICCV 2017)* [51] | RGB | 83.85 | 76.23 | 71.11 | 17.37 | 24.05 | 28.79 | 53.93 | 46.99 | 42.62 | 32.47 | 37.43 | 41.59 |
| CorrWise (CVPR 2022)* [52] | RGB | 92.8 | N/A | **83.9** | 8.5 | N/A | 15 | 82 | N/A | 66.7 | 17.2 | N/A | 25.9 |
| OPT (CVPR 2022)* [53] | RGB | 94.54 | 86.89 | 80.4 | 6.46 | 12.5 | 17.83 | 82.71 | 69.5 | 61.09 | 12.34 | 20.29 | 26.35 |
| DMVFN (CVPR 2023)* [13] | RGB | 95.73 | **89.24** | 83.45 | 5.58 | 10.47 | 14.82 | **88.53** | **78.01** | **70.52** | 10.74 | 19.27 | 26.05 |
| Ours | RGB | 94.85 | 87.82 | 82.11 | **4.13** | **8.12** | **11.70** | 87.70 | 77.15 | 69.72 | **9.50** | **16.94** | **22.90** |

In Table 5, we compare the proposed method with existing methods on the Cityscapes and KITTI datasets. Following previous research, we report the Multi-scale Structural Similarity Index Measure (MS_SSIM) and LPIPS of the first future frame ($t+1$), the average numbers of the next three future frames ($t+3$) and the average numbers of the next five future frames ($t+5$).

The UCF Sports dataset features sports scenes and thus contains a large amount of fast movements and motion blurs. The leading performance in both Table 3 and 4 validates that the proposed motion graph helps the network better interpret the motion in the input video sequence and facilitate more accurate prediction. Meanwhile, the qualitative result in Figure 5 demonstrates the method's ability to capture and restore intricate image details during the prediction.

For KITTI and Cityscapes datasets, videos are captured from outdoor, large-scale traffic scenarios through cameras with perspective projection, which results in drastic object distortions on both sides of images as well as unstable lighting conditions. Our method matches SOTA performance in terms of quantitative evaluation (see Table 5). In Figure 6, we conducted qualitative comparison on these datasets with two SOTA methods, OPT [53] and DMVFN [13], which are all optical-flow-based.

We noticed that as a non-generative model, our proposed video prediction system may face challenges with occlusions that require the generation of unseen objects. However, it excels in scenarios with scenarios involving occlusions of known objects, as showcased in white walls of the third column in Figure 6. Notably, in scenes with partial obstructions—such as the white wall behind the cyclist, our system adeptly employs multiple motion vectors per pixel to reconstruct occluded areas. This feature
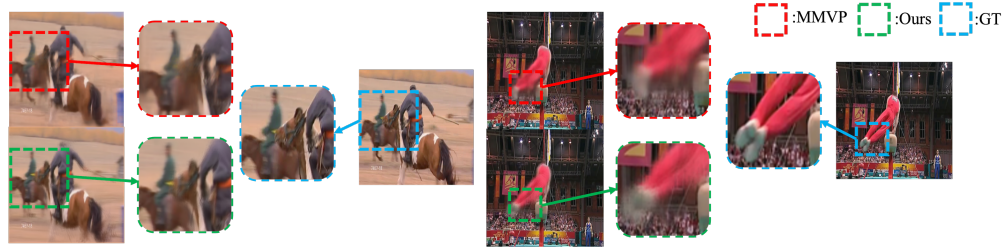
Figure 5: On the UCF Sports dataset, our method recovers richer image details than MMVP [2].



Figure 6: Qualitative comparisons with OPT [53] and DMVFN [13]. Our method maintains the object structures better than both methods while holding a higher motion prediction accuracy.

also supports precise management of object expansion due to perspective projection, as exemplified by the green car in Figure 6's first column.

Moreover, our approach is adept at managing objects exiting the camera's view. By explicitly modeling the motion of image patches, the motion graph predicts when features are about to leave the scene. Thus, any image patches projected to move out of view are not included in the final prediction. This is demonstrated in the 1st, 3rd, and 4th columns of Figure 6, where objects like a cyclist and the front of a blue truck are shown as moving out of frame. Our method successfully and uniquely captures and represents these movements, unlike how the other methods evaluated.

To assess the compactness of the motion graph, we conduct a model consumption analysis. We compare the proposed method with SOTA methods on each dataset in terms of the model size and maximum running GPU memory (the system configurations are identical to the models that generate the numbers in the corresponding tables above). Examination of Table 6 reveals that our method shows robust advantages than SOTA methods in minimizing model consumption and saving resources.

Table 6: Model consumption analysis on three datasets, compared with the SOTA methods.

| Dataset | Image Resolution | Input Frame | Method | Model Size | GPU Memory |
|---------|-----------------|-------------|--------|------------|------------|
| UCF Sports | $512 \times 512$ | 4 | MMVP [2] | 2.79M | 4.53GB |
| | | | Ours | **0.60**M | **2.38**GB |
| KITTI | $256 \times 832$ | 2 | DMVFN [13] | 3.56M | 3.79GB |
| | | | Ours | **0.97**M | **0.97**GB |
| Cityscapes | $512 \times 1024$ | 2 | DMVFN [13] | 3.56M | 7.41GB |
| | | | Ours | **0.97**M | **1.25**GB |

## 4.2 Ablation Studies

We conduct ablation studies on three aspects of the UCF Sports MMVP split: i) the choice of $k$, which defines three attributes of the system, i.e., the number of dynamic vectors each node initially embeds,

the number of temporal forward/backward edges for each node, and the number of dynamic vectors to be decoded from the upsampled motion features; ii) the number of views used for multi-view motion graph construction; and iii) the composition of the node's initial features.

Table 7: Ablation study on the value of $k$.

| $k$ | SSIM ↑ | PSNR ↑ | LPIPS ↓ | Memory ↓ |
|---|---|---|---|---|
| 1 | 0.9199 | 29.84 | 0.0868 | 0.98G |
| 5 | 0.9273 | 30.19 | 0.0966 | 1.61G |
| 8 | 0.9285 | 30.25 | 0.0934 | 2.07G |
| 10 | 0.9314 | 30.49 | 0.0823 | 2.38G |
| 20 | 0.9319 | 30.58 | 0.0809 | 3.96G |

Table 8: Ablation study on the number of motion graph views and on the location feature $\mathbf{f}_{loc}$.

| View Number | $\mathbf{f}_{loc}$ | SSIM ↑ | PSNR ↑ | LPIPS ↓ |
|---|---|---|---|---|
| 1 | ✓ | 0.9058 | 28.53 | 0.1205 |
| 2 | ✓ | 0.9158 | 29.47 | 0.1064 |
| 4 | ✓ | 0.9314 | 30.52 | 0.0832 |
| 4 | ✗ | 0.9247 | 30.27 | 0.1008 |

**The choice of** $k$: Adjustments to the value of $k$ and subsequent evaluations reveal a consistent, monotonic increase in both performance and GPU memory usage, as documented in Table 7. Notably, the gain rate diminishes as $k$ increases. This observation aligns with the intuition that increasing the value of $k$ will bring more temporal correspondences to the system. However, once the motion information nears saturation, the incremental benefit of further increasing $k$ becomes less significant. More ablation studies related to $k$ can be found in Section A.4.

**The number of graph views**: As Section 3.2 mentioned, using multi-view graphs can enhance motion prediction accuracy. To validate this claim, we reduce the number of views used in the system and test the system on the UCF Sports dataset. The first three rows in Table 8 show that increasing the number of graph views indeed improves the system performance.

**Location feature** $\mathbf{f}_{loc}$: We incorporate the tendency feature $\mathbf{f}_{tdc}$ (directly related to motion) and a location feature $\mathbf{f}_{loc}$ to initialize the node features. This inclusion hypothesizes that an image pixel's motion patterns may also be influenced by its spatial position. Empirical evidence supporting this hypothesis is documented in the last two rows of Table 8. There is a notable decline in performance when the location feature $\mathbf{f}_{tdc}$ is excluded from the node feature initialization. In Section A.7, we further visualize the location features and observe that such feature may reflect the camera projection pattern, providing substantial information to the following motion reasoning task.

## 5    Conclusion & Limitation

This work presents the motion graph for video prediction and a novel pipeline built upon it. We focus on balancing *representative ability* and *compactness* to optimize efficiency. The motion graph encapsulates complex motion information hidden in video sequences into a more manageable format, and achieves encouraging results: it matched or outperformed SOTA on three well-known datasets with a considerably smaller model size and less GPU running memory. Beyond the results, the motion graph and its associated video prediction pipeline set a foundation for further research and optimization in the field of video prediction, thus suggesting a promising direction for researchers seeking to balance performance and resource efficiency in the evolving domain of video processing.

**Limitation.** There is no significantly improvement in terms of the inference speed. The current version of our model is slower than DMVFN, which is specifically optimized for this aspect. Future efforts will focus on accelerating inference while maintaining the model's compactness and efficiency. Additionally, video prediction remains particularly challenging in scenarios involving sudden or unpredictable motions, which our system occasionally fails to capture, as highlighted in Appendix A.6. These instances, where the model struggles with abrupt actions not readily discernible from the input frames, underscore the need for further enhanced video understanding abilities. Addressing these challenges presents a valuable direction for future research.

## References

[1] Zhangyang Gao, Cheng Tan, Lirong Wu, and Stan Z Li. Simvp: Simpler yet better video prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3170–3180, 2022.

[2] Yiqi Zhong, Luming Liang, Ilya Zharkov, and Ulrich Neumann. Mmvp: Motion-matrix-based video prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4273–4283, 2023.

[3] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28, 2015.

[4] Yunbo Wang, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and Philip S Yu. Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms. *Advances in neural information processing systems*, 30, 2017.

[5] Yunbo Wang, Zhifeng Gao, Mingsheng Long, Jianmin Wang, and S Yu Philip. Predrnn++: Towards a resolution of the deep-in-time dilemma in spatiotemporal predictive learning. In *International Conference on Machine Learning*, pages 5123–5132. PMLR, 2018.

[6] Yunbo Wang, Jianjin Zhang, Hongyu Zhu, Mingsheng Long, Jianmin Wang, and Philip S Yu. Memory in memory: A predictive neural network for learning higher-order non-stationarity from spatiotemporal dynamics. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9154–9162, 2019.

[7] Haixu Wu, Zhiyu Yao, Jianmin Wang, and Mingsheng Long. Motionrnn: A flexible model for video prediction with spacetime-varying motions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15435–15444, 2021.

[8] Agrim Gupta, Stephen Tian, Yunzhi Zhang, Jiajun Wu, Roberto Martín-Martín, and Li Fei-Fei. Maskvit: Masked visual pre-training for video prediction. *arXiv preprint arXiv:2206.11894*, 2022.

[9] Shuliang Ning, Mengcheng Lan, Yanran Li, Chaofeng Chen, Qian Chen, Xunlai Chen, Xiaoguang Han, and Shuguang Cui. Mimo is all you need: a strong multi-in-multi-out baseline for video prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 1975–1983, 2023.

[10] Mingzhen Sun, Weining Wang, Xinxin Zhu, and Jing Liu. Moso: Decomposing motion, scene and object for video prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18727–18737, 2023.

[11] Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. Decomposing motion and content for natural video sequence prediction. *arXiv preprint arXiv:1706.08033*, 2017.

[12] Xiaojie Gao, Yueming Jin, Qi Dou, Chi-Wing Fu, and Pheng-Ann Heng. Accurate grid keypoint learning for efficient video prediction. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5908–5915. IEEE, 2021.

[13] Xiaotao Hu, Zhewei Huang, Ailin Huang, Jun Xu, and Shuchang Zhou. A dynamic multi-scale voxel flow network for video prediction. *arXiv preprint arXiv:2303.09875*, 2023.

[14] Lluis Castrejon, Nicolas Ballas, and Aaron Courville. Improved conditional vrnns for video prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7608–7617, 2019.

[15] Zheng Chang, Xinfeng Zhang, Shanshe Wang, Siwei Ma, and Wen Gao. Strpm: A spatiotemporal residual predictive model for high-resolution video prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13946–13955, 2022.

[16] Zheng Chang, Xinfeng Zhang, Shanshe Wang, Siwei Ma, and Wen Gao. Stip: A spatiotemporal information-preserving and perception-augmented model for high-resolution video prediction. *arXiv preprint arXiv:2206.04381*, 2022.

[17] Yunbo Wang, Lu Jiang, Ming-Hsuan Yang, Li-Jia Li, Mingsheng Long, and Li Fei-Fei. Eidetic 3d lstm: A model for video prediction and beyond. In *International conference on learning representations*, 2018.

[18] Jun-Ting Hsieh, Bingbin Liu, De-An Huang, Li F Fei-Fei, and Juan Carlos Niebles. Learning to decompose and disentangle representations for video prediction. *Advances in neural information processing systems*, 31, 2018.

[19] Xi Ye and Guillaume-Alexandre Bilodeau. Vptr: Efficient transformers for video prediction. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 3492–3499. IEEE, 2022.

[20] Rohit Girdhar and Kristen Grauman. Anticipative video transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 13505–13515, 2021.

[21] Nal Kalchbrenner, Aäron Oord, Karen Simonyan, Ivo Danihelka, Oriol Vinyals, Alex Graves, and Koray Kavukcuoglu. Video pixel networks. In *International Conference on Machine Learning*, pages 1771–1779. PMLR, 2017.

[22] Xiaodan Liang, Lisa Lee, Wei Dai, and Eric P Xing. Dual motion gan for future-flow embedded video prediction. In *proceedings of the IEEE international conference on computer vision*, pages 1744–1752, 2017.

[23] Yung-Han Ho, Chuan-Yuan Cho, Wen-Hsiao Peng, and Guo-Lun Jin. Sme-net: Sparse motion estimation for parametric video prediction through reinforcement learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10462–10470, 2019.

[24] Hang Gao, Huazhe Xu, Qi-Zhi Cai, Ruth Wang, Fisher Yu, and Trevor Darrell. Disentangling propagation and generation for video prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9006–9015, 2019.

[25] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 541–556. Springer, 2020.

[26] Maosheng Ye, Tongyi Cao, and Qifeng Chen. Tpcn: Temporal point cloud networks for motion forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11318–11327, 2021.

[27] Roger Girgis, Florian Golemo, Felipe Codevilla, Martin Weiss, Jim Aldon D'Souza, Samira Ebrahimi Kahou, Felix Heide, and Christopher Pal. Latent variable sequential set transformers for joint multi-agent motion prediction. *arXiv preprint arXiv:2104.00563*, 2021.

[28] Maosen Li, Siheng Chen, Yangheng Zhao, Ya Zhang, Yanfeng Wang, and Qi Tian. Dynamic multiscale graph neural networks for 3d skeleton based human motion prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 214–223, 2020.

[29] Lingwei Dang, Yongwei Nie, Chengjiang Long, Qing Zhang, and Guiqing Li. Msr-gcn: Multi-scale residual graph convolution networks for human motion prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11467–11476, 2021.

[30] Chongyang Zhong, Lei Hu, Zihao Zhang, Yongjing Ye, and Shihong Xia. Spatio-temporal gating-adjacency gcn for human motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6447–6456, 2022.

[31] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016.

[32] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020.

[33] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2472–2481, 2018.

[34] Simon Niklaus and Feng Liu. Softmax splatting for video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5437–5446, 2020.

[35] Simon Niklaus, Ping Hu, and Jiawen Chen. Splatting-based synthesis for video frame interpolation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 713–723, 2023.

[36] Mikel D Rodriguez, Javed Ahmed, and Mubarak Shah. Action mach a spatio-temporal maximum average correlation height filter for action recognition. In *2008 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2008.

[37] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[38] Yue Wu, Rongrong Gao, Jaesik Park, and Qifeng Chen. Future video synthesis with object motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5539–5548, 2020.

[39] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[40] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[41] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. *arXiv preprint arXiv:1511.05440*, 2015.

[42] Alex X Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018.

[43] Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H Campbell, and Sergey Levine. Stochastic variational video prediction. *arXiv preprint arXiv:1710.11252*, 2017.

[44] Yong-Hoon Kwon and Min-Gyu Park. Predicting future frames using retrospective cycle gan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1811–1820, 2019.

[45] Wei Yu, Yichao Lu, Steve Easterbrook, and Sanja Fidler. Efficient and information-preserving future frame prediction and beyond. 2020.

[46] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. *arXiv preprint arXiv:1808.06601*, 2018.

[47] Junting Pan, Chengyu Wang, Xu Jia, Jing Shao, Lu Sheng, Junjie Yan, and Xiaogang Wang. Video generation from single semantic label map. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2019.

[48] Yue Wu, Rongrong Gao, Jaesik Park, and Qifeng Chen. Future video synthesis with object motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5539–5548, 2020.

[49] Xinzhu Bei, Yanchao Yang, and Stefano Soatto. Learning semantic-aware dynamics for video prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 902–912, 2021.

[50] William Lotter, Gabriel Kreiman, and David Cox. Deep predictive coding networks for video prediction and unsupervised learning. *arXiv preprint arXiv:1605.08104*, 2016.

[51] Ziwei Liu, Raymond A Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *Proceedings of the IEEE international conference on computer vision*, pages 4463–4471, 2017.

[52] Daniel Geng, Max Hamilton, and Andrew Owens. Comparing correspondences: Video prediction with correspondence-wise losses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3365–3376, 2022.

[53] Yue Wu, Qiang Wen, and Qifeng Chen. Optimizing video prediction via video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17814–17823, 2022.

[54] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[55] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[56] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[57] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Atlanta, GA, 2013.

[58] Vikram Voleti, Alexia Jolicoeur-Martineau, and Chris Pal. Mcvd-masked conditional video diffusion for prediction, generation, and interpolation. *Advances in neural information processing systems*, 35:23371–23385, 2022.

[59] Kangfu Mei and Vishal Patel. Vidm: Video implicit diffusion models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 9117–9125, 2023.

[60] Zhicheng Zhang, Junyao Hu, Wentao Cheng, Danda Paudel, and Jufeng Yang. Extdm: Distribution extrapolation diffusion model for video prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19310–19320, 2024.

# A  Appendix

## A.1  Implementation Details

We implement the video prediction system using PyTorch [54] and conduct end-to-end training on a single NVIDIA A100 GPU. We use AdamW optimizer [55] during the training. The initial learning rate is set to $1e^{-3}$ and decayed to $1e^{-5}$ following a cosine decay scheduler [56]. There are only a few hyperparameters to adjust for the system when training on different datasets. The adjustments are mainly based on the resolution of the frame. Those hyper-parameters include i) image feature length (Image feat.), which is the parameter for the image encoder; ii) Tendency feature length (Tendency feat.), which is the length of the tendency feature in node initialization; iii) location feature length (Location feat.), which is fixed to 4 for all datasets, indicating the length of the location feature in node initialization; iv) the number of graph views, indicating view number in motion graph feature learning; v) $k$, indicates the number of the dynamic vectors embedded in each node, the number of the temporal edges per node and the output dynamic vectors per pixel; vi) training epoch is the training related parameters; vii) the reconstruction loss, which follows the popular setting of the SOTA methods on each dataset. In Table 9, we demonstrate the hyper-parameter setting for each dataset.

Please note that we did not especially tune the parameters for each dataset. When adjusting the parameters, we consider more about the training efficiency instead of the performance. Therefore, our setting is likely *not* the optimal choice. For example, in DMVFN [13], the training on Cityscapes and KITTI are both 300 epochs, we observe that our system can achieve comparable performance with only 100 and 200 epochs respectively, we thus stay with this configuration.

| Dataset | Image feat. | Tendency feat. | Location feat. | Number of Graph views | $k$ | Epoch | Loss |
|---|---|---|---|---|---|---|---|
| UCF Sports | 16 | 16 | 4 | 4 | 10 | 300 | MSE |
| Cityscapes | 16 | 32 | 4 | 4 | 10 | 100 | L1 + Lpips |
| KITTI | 16 | 32 | 4 | 4 | 8 | 200 | L1 + Lpips |

Table 9: Hyper-parameter setting for each dataset.

## A.2  Network Architecture

The proposed video prediction system includes three major components, the image encoder, the motion graph interaction module, and the motion upsampler. Here we demonstrate the detailed architecture of each component for reproduction needs.

**Image Encoder**: Figure 7 shows the inner structure of the image encoder in the proposed system. $C_{img}$ is related to the image feature length in Table 9. Each convolution layer will come with a Leaky ReLU layer [57] as the activation layer.
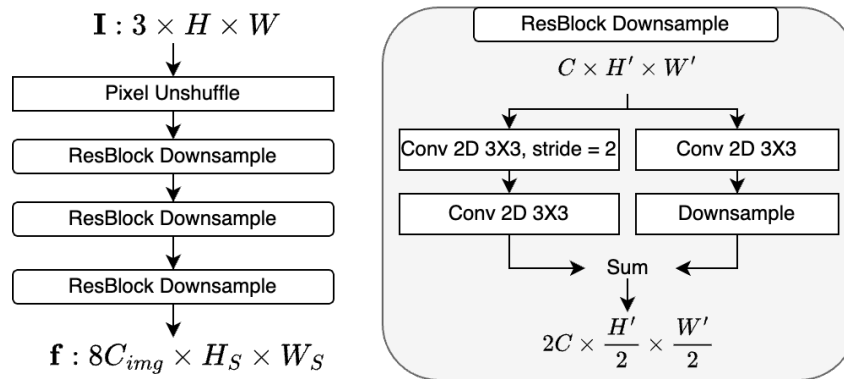


Figure 7: Architecture of the image encoder

**Motion Graph Interaction Module** In Figure 8 we demonstrate the inner structure of the spatial and temporal message passing in the motion graph interaction module. $C_{node}$ equals the sum of the tendency feature length and the location feature length in Table 9.

**Spatial:**

$$V : N \times C_{node} \xrightarrow{Reshape} V : C_{node} \times H_S \times W_S \rightarrow \boxed{\text{Conv2D 3X3}} \xrightarrow{Reshape} V' : N \times C_{node}$$

**Temporal:**

$$V_{pred} : N_{pred} \times C_{node} \longrightarrow \boxed{\text{Linear Projection}} \rightarrow V_{pred} : N_{pred} \times C_{node}$$
$$\downarrow \times W_{pred} : N_{pred} \times 1$$
$$V_{suc} : 1 \times C_{node} \longrightarrow \oplus \rightarrow \boxed{\text{Linear Layer}} \rightarrow V'_{suc} : 1 \times C_{node}$$
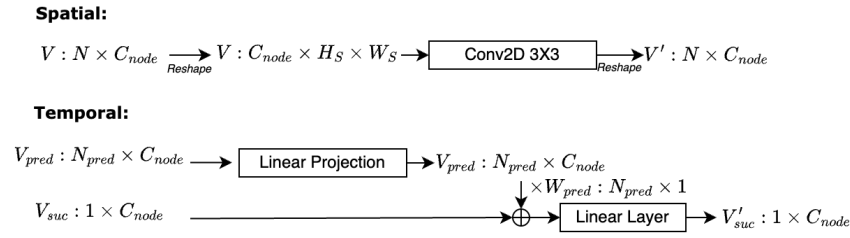
Figure 8: Inner structure of spatial and temporal block in motion graph interaction module

**Motion Upsampler** Figure 9 illustrates the inner structure of the motion upsampler as well as the motion decoder. The implementation of the decoder is a single 2D convolution layer with a kernel size of 1.
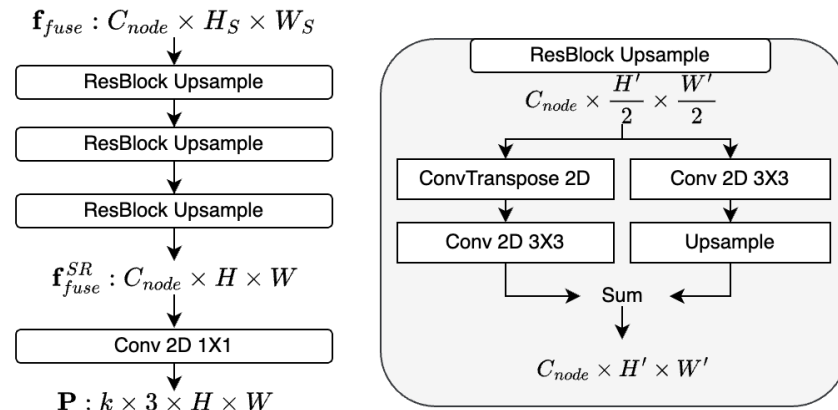


Figure 9: Inner structure of the motion upsampler and the motion decoder.

## A.3 Additional Quantitative Evaluation

For comparison convenience of the future works, we list the extra metrics of cityscapes and KITTI datasets in Table 10, i.e. ssim and psnr.

Table 10: Evaluation on Cityscapes [39] and KITTI [37] datasets. "RGB", "F", "S" and "I" denote video frames, optical flow, semantic map, and instance map; $t + 3$ and $t + 5$ respectively indicate the average performance of the next 3 and 5 frames. Results with * are copied from [13].

| Method | Input | Cityscapes | | | | | | KITTI | | | | | |
| | | SSIM($\times 10^{-2}$)↑ | | | PSNR↑ | | | SSIM($\times 10^{-2}$)↑ | | | PSNR↑ | | |
| | | $t+1$ | $t+3$ | $t+5$ | $t+1$ | $t+3$ | $t+5$ | $t+1$ | $t+3$ | $t+5$ | $t+1$ | $t+3$ | $t+5$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DMVFN (CVPR 2023)* [13] | RGB | 87.45 | 78.75 | 63.95 | 28.81 | 25.47 | 23.61 | 72.69 | 62.52 | 57.05 | 22.74 | 20.16 | 18.70 |
| Ours | RGB | 92.30 | 86.51 | 83.16 | 28.43 | 25.39 | 23.80 | 77.91 | 69.55 | 65.05 | 21.71 | 19.44 | 18.18 |

## A.4 Extensive Ablation Study

In this section, we add two ablation studies to help the audience better interpret the design of the motion graph.

**Number of the predicted dynamic vectors per pixel:** In the proposed system, we set the number of the predicted dynamic vectors per pixel to $k$, which is identical to the number of the dynamic vectors embedded by each node and the temporal edge of each node. This design ensures the flexibility of the predicted motion to have multiple modes compared to the optical-flow-based method which

only allows each pixel to have a single future motion. The comparison between the first two rows of Table 11 showcase that allowing multiple predicted dynamic vectors can largely improve the performance. Meanwhile, if we control the number of the predicted dynamic vectors, as demonstrate by the comparison between the first and third row of the Table 11, we see that when the motion graph embeds more past motion modes, the performance will also has significant improvements.

| $k$ | Predicted Vectors # | Full set | | | Hard (SSIM < 0.6) | | | Memory ↓ |
|---|---|---|---|---|---|---|---|---|
| | | SSIM ↑ | PSNR ↑ | LPIPS ↓ | SSIM ↑ | PSNR ↑ | LPIPS ↓ | |
| 1 | 1 | 0.8742 | 26.34 | 0.1527 | 0.5394 | 17.34 | 0.5032 | 0.98G |
| 1 | 10 | 0.9199 | 29.87 | 0.1042 | 0.6408 | 19.44 | 0.3706 | 1.97G |
| 10 | 1 | 0.9212 | 30.00 | 0.0877 | 0.6546 | 19.63 | 0.3261 | 1.38G |

Table 11: Ablation study on the number of the predicted vectors. The experiments are conducted on UCF Sports MMVP splits. The listed results are from the models trained for 100 epochs (models were all trained for 300 epochs in the main manuscript).

**Motion Graph Interaction Module** The design of the motion graph interaction module are following the intuition that both spatial connection and temporal connection should benefit the graph learning. Here we also show the experimental results in Table 12 that both spatial and backward edges are beneficial to the final performance.

| Spatial | Backward | PSNR ↑ | MS-SSIM ×100 ↑ | LPIPS×100 ↓ |
|---|---|---|---|---|
| ✕ | ✕ | 21.55 | 87.06 | 9.85 |
| ✓ | ✕ | 21.64 | 87.25 | 9.83 |
| ✓ | ✓ | 21.71 | 87.70 | 9.50 |

Table 12: Ablation study on graph interaction module. The experiments are conducted on KITTI and metrics show evaluation on the $t + 1$ results.

## A.5 Extensive Discussion on Experiment setting

Our research on video prediction has identified key differences between systems designed for short-term and long-term predictions. Short-term systems typically use fewer frames to predict the immediate next few frames, while long-term systems are tasked with forecasting an extensive sequence of future frames. The design logic and objectives of these systems, thus, vary significantly. Table 13 demonstrates the differences with more details. It is easy to notice that the recent works of

| | Short-term Video Prediction | Long-term Video Prediction |
|---|---|---|
| Prediction Length | Limited, usually one frame | Much longer, $\gg 10$ frames |
| Video Resolution | Up to 4K | Usually smaller, up to $256 \times 256$ |
| Common Dataset | UCF 101, UCF Sports, KITTI, Cityscapes, SJTU4k, Vimeo, DAVIS | KTH, moving MNIST, BAIR, cropped Cityscapes |
| Recent works | SIMVP[1], MMVP[2], STRPM[15], STIP[16], DMVFN[13] | MVCD[58], MaskViT[8],VIDM[59],ExtDM[60] |
| Objectives | High resolution videos, pixel-level accuracy, real-time application | Conditional video generation, semantic-level accuracy |

Table 13: Experiment setting comparison between short-term and long-term video prediction task

long-term video prediction usually involve with diffusion-based generative models, which is designed to feed the need of predicting long videos. The evaluation of such methods usually emphasize on if the predicted video is semantically correct given the input video frame(s). While in this study, we emphasize our method's advanced motion modeling and significant reduction in computational costs, essential for short-term prediction of high-resolution videos. Our system is tailored for short-term video prediction, with evaluations conducted along this line of work.

We plan to further exploit the motion graph's potential as an efficient motion representation tool and develop advanced, motion graph-based systems for long-term video generation in our next work.

## A.6 Failure case demonstration

The video prediction is always a challenging problem. Especially for those video sequences with abrupt motion which can be hardly indicated by the previous video frames. The proposed method

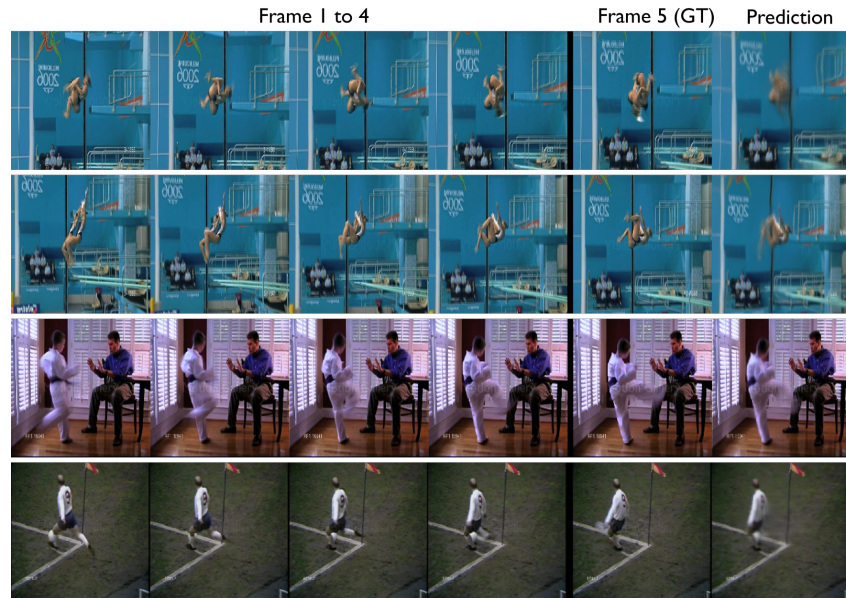Frame 1 to 4        Frame 5 (GT)    Prediction



Figure 10: Failure cases in UCF Sports Dataset

formulates the video prediction as a motion prediction problem and outperforms most of the existing methods by using motion graph to better capture the motion hints from the input frames. However, when evaluating the qualitative results, we still find some failure cases that require additional research efforts to solve. In Figure 10, we showed typical failure cases in UCF Sports dataset. We notice that most of the failures cases are in the action of kicking and diving, which usually include fast, unpredictable motion that requires stronger video understanding capability of the model.

### A.7 Node Feature Visualization

To better understand the initialization of the node embedding, here we visualize the tendency feature and the location feature. We first extract the tendency feature and location feature of each node in the motion graph and apply a K-means clustering to the extracted features. For the tendency feature, we set the cluster number to 2; and for the location feature, we set the cluster number to 4 for better visualization.

From Figure 11, we see that using the learned tendency feature, the system should be able to distinguish the dynamic areas from the static areas. If we further enlarge the cluster number, we can see more clearly that the tendency features embed the different motion patterns of each feature patch in the frame. For the location feature, in the paper, we have shown that removing the location feature from the node initialization will result in a performance drop. From Figure 12 we observe that the location feature may contain information that is related to the camera projection mode. For cityscapes and the KITTI, which use wide-range cameras, the clustering pattern of the location feature is very different from the UCF Sports whose projection mode is possible to be orthogonal projection.
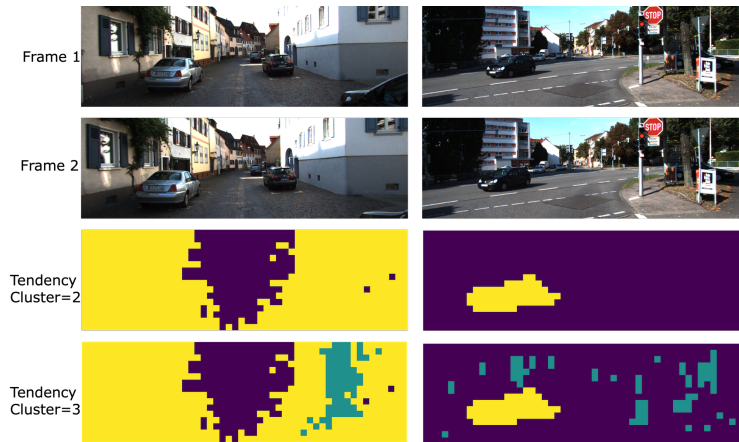
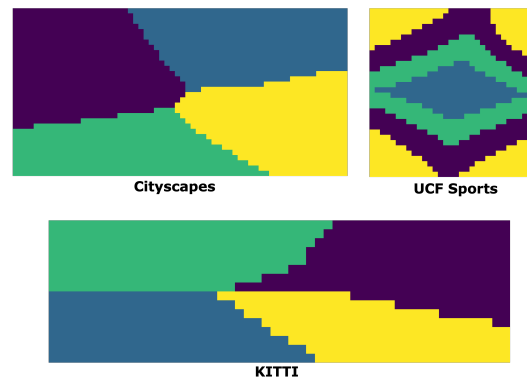Figure 11: Tendency feature visualization using KITTI dataset



Figure 12: Locaiton feature visualization on three datasets.

# NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Section 3 describes the detailed methodology and the Section 4 validates the claim.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: Please see Section 5

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Please see Section 4, Appendix A.1 and Appendix A.2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The data is all public and the code is provided in the supplementary zip file.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Please see Section 4 and Appendix A.1.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Please see Appendix A.1

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.

- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Please see Appendix A.1

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer:[Yes]

    Justification: Please see Appendix A.1, A.2 and the code comments provided in the supplementary file.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

   Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

   Answer: [NA]

   Guidelines:

   - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
   - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
   - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
   - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.