Skeleton-Graph: Long-Term 3D Motion Prediction From 2D Observations Using Deep Spatio-Temporal Graph CNNs

Abduallah Mohamed Huancheng Chen Zhangyang Wang Christian Claudel
The University of Texas At Austin

Abstract

Several applications such as autonomous driving, augmented reality and virtual reality require a precise prediction of the 3D human pose. Recently, a new problem was introduced in the field to predict the 3D human poses from observed 2D poses. We propose Skeleton-Graph, a deep spatio-temporal graph CNN model that predicts the future 3D skeleton poses in a single pass from the 2D ones. Unlike prior works, Skeleton-Graph focuses on modeling the interaction between the skeleton joints by exploiting their spatial configuration. This is being achieved by formulating the problem as a graph structure while learning a suitable graph adjacency kernel. By the design, Skeleton-Graph predicts the future 3D poses without divergence in the long-term, unlike prior works. We also introduce a new metric that measures the divergence of predictions in the long term. Our results show an FDE improvement of at least 27% and an ADE of 4% on both the GTA-IM and PROX datasets respectively in comparison with prior works. Also, we are 88% and 93% less divergence on the long-term motion prediction in comparison with prior works on both GTA-IM and PROX datasets. Code is available at https://github.com/abduallahmohamed/Skeleton-Graph.git.

1. Introduction

An accurate 3D pose prediction model is vital for several applications. In intersection management and autonomous vehicles, one can prevent an accident based on the 3D poses of pedestrians [10, 9, 30]. In Virtual and Augmented Reality (VR/ AR) the predictions of the 3D pose help in deepening the immersive experience [28]. Where in drones and autonomous driving, 3D pose prediction helps in accurate maneuver and motion planning through the environment [14, 42, 15, 11]. The process of obtaining the 2D poses is quite inexpensive in comparison to the process of obtaining the 3D poses. The 2D process does not require special sensors such as depth sensors installed on the device.

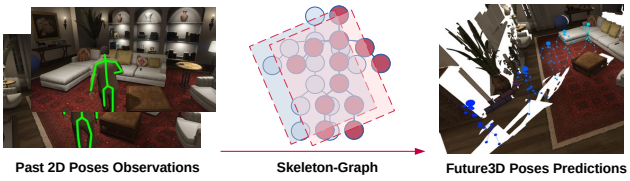

Figure 1: Skeleton-Graph given an input of 2D spatio-temporal graph of observed human skeleton poses. Then, in a single pass it predicts the next 3D poses.

Thus it is cheaper to obtain the 2D poses from an ordinary vision sensor [3]. However, 2D poses are not suitable for the increasing trends of requirements in a new range of applications such as the ones mentioned before [32, 31]. Thus it is tempting to develop models that directly obtain 3D poses from 2D ones, saving the need for expensive equipment and upscaling to the new trends.

The recent work of [6] introduced a new long-term trajectory prediction problem that focuses on the concept of obtaining 3D poses from 2D ones. The introduced problem goal is to predict the future expensive 3D motion trajectories from the cheaply obtained 2D observed motion. In their work, a deep model called GPP-Net was introduced to address this problem. It is a three-stage deep model that uses several concepts such as Variational Auto Encoders (VAEs) and tailored stages for both paths and poses predictions. Beside GPP-Net, other 3D human motion estimation models such as TR [36], VP [27], LTD [38] were evaluated on this problem.

By examining the results and the architectures of these prior works, we found that three main design components were used separately in each prior work to enhance the results but not collectively in one work. The first component is the exploitation of the spatial configuration of the skeleton explicitly through the deep architecture itself [38]. The skeleton spatial configuration includes useful information the leads to better predictions. The joints correlate with each other in terms of the angles and distances between them.

This kind of information when introduced to a deep model it will constraints the model output not to produce random points that are far away from the ground truth. The second component is the usage of the vision signal [6]. The vision signal of the observed sequence contains information such as the objects in the environment and the geometry of the scene. This information helps in resolving the ambiguity in some prediction scenarios. For example, it is not valid to predict a skeleton standing on a table or a couch. The third and last component is to avoid the use of deep recurrent architectures. Recurrent architectures in long-term prediction problems tend to accumulate prediction errors from a step to the next step. We noticed this behavior in prior works [27, 6, 38] that used these recurrent models.

Thus we introduce Skeleton-Graph in which we combine these three design components in one work overcoming the shortcoming of the prior works. First, to use the spatial configuration information of the skeleton we model the problem as a spatio-temporal graph end to end. We rely on the adjacency matrix to encode the relationship between the skeleton joints. Instead of a fixed adjacency matrix, we let our model learn it and analyze it in our ablation study. Secondly, we utilize the vision signal by fusing it into our model. Thus, we utilize the context information to enhance our results. We found out that the visual signal in some cases enhances short-term predictions. Lastly, to avoid divergence over the long-term we use a full CNN approach end to end without any recurrent behavior. This led to better long-term predictions in comparison with prior works. We introduce a new metric that measures the divergence over the prediction horizon to quantitatively judge this criterion of prediction stability.

This work is organized as follows, we start with the literature review of related 2D and 3D pose estimation methods, as well as deep graph models and trajectory prediction methods. Then, we follow up with the problem formulation and description of Skeleton-Graph method. Next, we discuss the problem of inconsistency in the prediction of the 3D skeleton poses and introduce a new objective that ensures the predicted 3D skeleton looks natural. We analyze the performance of our approach both quantitatively and qualitatively and discuss the prediction stability over the long term.

2. Related Work

2D pose estimation: With the resurgence of neural nets, data-driven prediction paradigms have become more dominant. DeepPose [35] was the first major paper that applied deep learning to human pose estimation. In this approach, pose estimation is formulated as a CNN-based regression problem towards body joints. The work of [34] generates heatmaps, describing the likelihood of the skeleton joints by running an image through multiple resolution banks in parallel to simultaneously capturing features at a variety of images scales. [26] Introduced a novel and intuitive architecture that consists of steps of pooling and upsampling layers to capture information at every scale. The previous approaches work well but were complicated in comparison with the next work. The work of [39] came up with a quite simple but efficient structure that consists of ResNet and few deconvolutional layers instead of the upsampling mechanism.

3D pose estimation: Recovering 3D pose from 2D RGB images is considered more difficult than 2D pose estimation, due to the larger 3D pose space and more ambiguities. The work of [22] built a framework that consists of a joint points regression task and point detection task. They train a network that directly regresses 3D points from an image. Later, [7] explored a simple architecture that reasons through intermediate 2D pose estimations instead of directly estimating 3D pose from an image. Further [24] explored the sources of error in 3D pose estimation. They doubted the source of error is either from 2D to 3D mapping or from the improper visual analysis of the scene. They concluded that lifting the ground truth 2D joints locations to 3D space is not the source of error and it is a relatively straightforward task but the visual analysis is the main issue. However, since these methods completely ignore the image context, the predicted human motion may not be consistent with the scene. To perceive the scene context in the pose estimation task, [13] exploits static 3D scene structure to better estimate human pose from monocular images considering environment constraints. A limitation of the current formulation is that it does not model scene occlusion. More recently, [6] formulates a new task of long-term 3D human motion prediction with scene context in terms of 3D poses and develops a novel three-stage computational framework that utilizes scene context for goal-oriented motion prediction.

Advances in deep graph models: Recent advances in deep graph CNNs [19] lead to a jump in the domain of pose and trajectory predictions [25, 16, 37, 5, 37] as graphs CNNs can exploit both the spatial configurations of the agents and their corresponding inner features. Several lines of work successfully classified human skeletons' actions by considering a spatio-temporal graph formulation of the skeleton [40, 17]. The recent work of [21] proposed a multi-scale deep graph network to predict 3D human motion from 3D history.

3. Problem Formulation and Technical Approach

Given a pair of observations of T time steps 2D human pose P_{2D} and T steps 2D images I^{2D} of the scene the goal is to predict the next $\tilde{T}$ steps 3D human poses P_{3D}. The human skeleton have J keypoints such that $P_{\text{2D}} \in \mathbb{R}^{J\times 2}$ and $P_{\text{3D}} \in \mathbb{R}^{J\times 3}$. Also, the pose prediction problem includes a path prediction problem within it. The path prediction problem is important because it is required to keep track of

the 3D poses and to warp or re-normalize them when needed. The position of the path is the center of the torso.

In what follows, we describe our approach to solve the problem at hand. We first start by modeling the problem as a spatio-temporal graph that represents the 2D motion of the human skeleton over time. Then we describe the Skeleton-Graph model wrapping with the loss function.

3.1. The formulation of the spatio-temporal graph

We start modeling the problem as a spatio-temporal skeleton graph. On each observed time-step t, $\{t \in \mathbb{Z} \mid 0 \leq t \leq T\}$ we define a graph $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$. Where $\mathcal{V}_t$ is the graph vertices at time step t. Each vertex will hold the pose P^t_{2D} information of the corresponding j, $\{j \in \mathbb{Z} \mid 0 \leq j < J\}$ skeleton key-point. The $\mathcal{E}_t$ is the collection of graph edges. The adjacency matrix $\mathcal{A}_t$ defines the weight values of the graph edges. In our formulation we set the weights to be one, later we describe in our model how we learn proper values for entries of $\mathcal{A}_t$. Now, we can represent our input as a spatio-temporal graph $\mathcal{SG}_{\text{input}} = \{\mathcal{G}^{\text{2D}}_t \mid t \in \mathbb{Z}, 0 \leq t \leq T\}$ and the output is $\{\tilde{P}^t_{\text{3D}} \mid t \in \mathbb{Z}, T < t \leq \tilde{T}\}$ which is the next 3D poses.

3.2. Skeleton-Graph model

From a top view, our model consists of four main components. The spatio-temporal graph CNN (SPGCNN) receives the spatio-temporal graph of the observed motion and generates a representation for it. The second component lies within the SPGCNN component. It is a CNNs that learns an adjacency matrix based on the temporal skeleton structure. The third component is the vision graph fusion. This component mixes both the graph representation from SPGCNN and the visual signal from the vision extractor. Lastly, the time-extrapolator CNN (TXCNN) receives this fused representation of both visual and graph and predicts the next 3D poses. In the upcoming section, we explain each component separately. These components can be seen in Figure 2. Our code is open-sourced and includes all the implementation details of the model. The supplementary materials include the fine implementation details of each component.

The spatio-temporal graph CNN (SPGCNN) Our design of this layer processes the graph vertices data which has the shape of $T \times J \times [x, y]$ in two steps. The first step is a spatial step where it applies CNNs over the graph nodes weighting the CNN kernel by the values of the adjacency matrix. This spatial step takes into account the connection between the skeleton nodes by exploiting the adjacency matrix. This aligns with our design goal of using this information to enhance the results. Then, a temporal step is applied to the tensor from the spatial step. This step is a simple CNNs but acts on the time T as a features channel. We refer to the work of [40] regarding more details about the spatio-temporal graph CNNs. The output of this layer is a graph embedding that represents the observed 2D skeleton motion over time and has the shape of $T \times J \times F$, where F is the learned features of each joint.

Learning a proper adjacency matrix The work of [25] proposed a kernel function to weigh the adjacency matrix that is tailored to their problem. Instead, in our approach we let the model discover the best weights of the adjacency matrix [38]. The adjacency matrix $\mathcal{A}$ has the dimensions of $T \times J \times J$, one can imagine it as a 2D image with T features channels. Thus, we use CNNs that take the temporal skeleton adjacency matrix as an input and learns a new one. Then, this learned adjacency $\tilde{\mathcal{A}}$ is used within the previous component of our model. In our ablation study, we show the benefit of using this learned matrix. The learned adjacency matrix $\tilde{\mathcal{A}}$ has the dimensions of $T \times J \times J$ but differs from the original one $\mathcal{A}$ in terms of the entries. This can be seen in Figure 6.

Vision graph fusion As we observe 2D images I^{2D}, using them could be beneficial to our model. We have two options, the first as in [6] to use the last observed image only I^{2D}_T. The intuition behind using the last image is that it is the nearest observation of what is to be predicted next. We also tested the idea of using the full sequence of the observed images $I^{\text{2D}}_{0,\dots,T}$ to use more visual context. Yet, empirically as we will show in the experiments section using the last image improves the path prediction while the whole sequence of images improves the pose prediction on one of the datasets. The vision feature extractor is a CNNs that is designed to down-size the images in terms of the spatial aspects (Width and height) to match the graph embedding. Then, both the image embedding and the graph embedding are concatenated. This can be seen in Figure 3. This simple concatenation gives the model the ability to learn a fused representation leveraging both visual and graph contexts. The final representation has the dimensions of $T \times J \times F$. This representation is to be used by the next layer to predict the next $\tilde{T}$ 3D poses.

The time-extrapolator CNN (TXCNN) This is the last step in Skeleton-Graph model. As we arrive at the embedding that represents the history of the 2D observations, we attempt to predict the next $\tilde{T}$ 3D steps. As in [25, 2, 23, 37, 41], using CNNs was proven to be better than recurrent models as it does not result in diverged predictions. The TXCNN receives an embedding of the history with the shape of $T \times J \times F$. The TXCNN treats the time as a features channel and through ordinary CNNs, it predicts the next $\tilde{T}$ 3D poses $\tilde{P}$ steps. Simply, it extrapolates the time into the future moving it from T observed steps to $\tilde{T}$ predicted steps. The final output of our model is $\tilde{T} \times J \times [x, y, z]$ which is the predicted 3D poses.

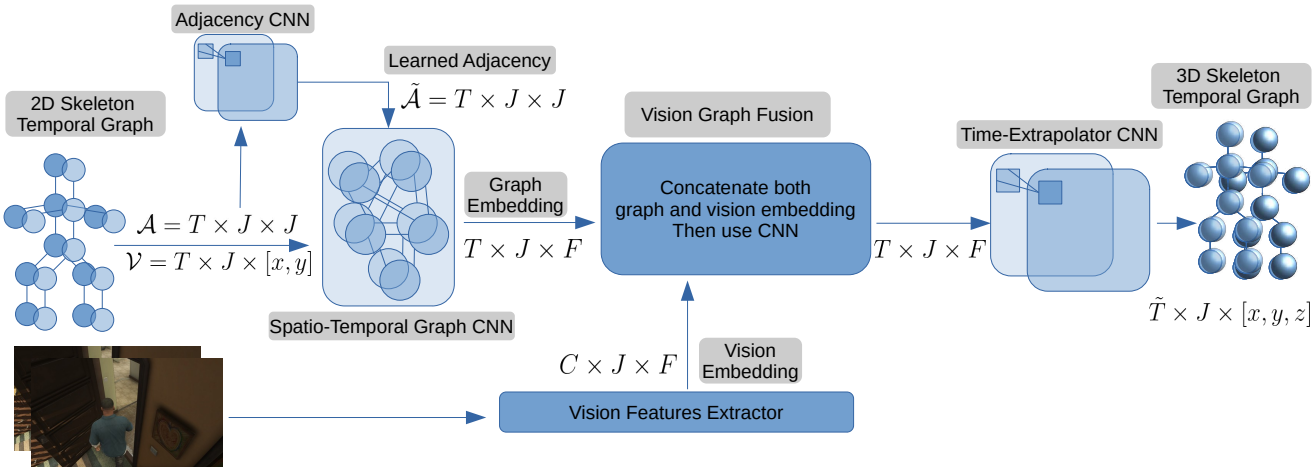

Figure 2: Skeleton-Graph model components. The model receives as an input a 2D skeleton temporal T graph poses $\mathcal{V}, \mathcal{A}$ and predicts the the next 3D skeleton temporal poses. The model can learn a suitable adjacency $\tilde{\mathcal{A}}$ matrix through the adjacency CNN. Also, it can use the observed visual signal in the form of a still image or a video. The time-extrapolator CNN is responsible for predicting the next $\tilde{T}$ temporal 3D poses, while the spatio-temporal graph CNN processes the 2D temporal graphs. $J, F, C, [x, y, z]$ are the number of the skeleton joints, the learned features dimensions, the vision features channels and the joint coordinates, respectively.

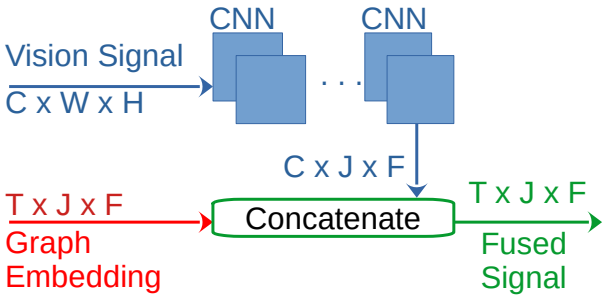

Figure 3: Vision graph fusion. C, W, H are the image channels, width and height.

4. The Skeleton Consistency Objective

During our experiments, we noticed that we have accurate results exceeding the state-of-the-art but the predicted 3D skeletons do not look natural. For example, a skeleton might be setting but the distance between the neck and body is awkward. Examples of these abnormal skeletons can be seen in Figure 4. Due to this, we introduce a loss function that forces the predicted skeletons to be more consistent. We call this loss Skeleton Consistency Loss (SCL) which enforces the correct bone length and angles between the joints in the predicted 3D poses. The skeleton consistency concept was introduced in prior works [4, 8, 29, 3, 33] in which they focus on the skeleton reconstruction problem not the 3D pose prediction problem. For the bone length, the prior approaches and ours use L_2 norm to force the correct bone length. For the angles between the joints, some used the rotation angles but we preferred to use the cosine similarity because of its well-defined range. As mentioned before our consistency loss is composed of two parts. The first part is a cosine similarity between each skeleton joint J in comparison with the predicted joints. It is defined as follows:

$$\text{SCL}_{\cos} = \frac{1}{\tilde{T}(J-1)} \sum^{t \in \tilde{T}} \sum_{i=1}^{J-1} \left\| \mathcal{C}\left(P_i^t, P_{i+1}^t\right) - \mathcal{C}\left(\tilde{P}_i^t, \tilde{P}_{i+1}^t\right) \right\|_1$$

Where $\mathcal{C}$ is the cosine similarity. The second part of the SCL is what enforces the reasonable bone length. It is defined as follows:

$$\text{SCL}_{L_2} = \frac{1}{\frac{\tilde{T}J(J-1)}{2}} \sum^{t \in \tilde{T}} \sum^{j \in J} \sum_{i=j+1}^{J} \left\| \left\| P_i^t - P_j^t \right\|_2 - \left\| \tilde{P}_i^t - \tilde{P}_j^t \right\|_2 \right\|_1$$

Thus, the SCL loss function will be defined as:

$$\mathcal{L}_{\text{SCL}} = \lambda_1 \text{SCL}_{\cos} + \lambda_2 \text{SCL}_{L_2} \quad (1)$$

Where λ_1 and λ_2 are weighting parameters. We found that setting both $\lambda_1 = 0.0005$ and $\lambda_2 = 0.1$ works the best.

Now, we can define the objective function that we train against:

$$\mathcal{L}_{\text{Skeleton-Graph}} = \frac{1}{\tilde{T}J} \sum_{t=1}^{\tilde{T}} \sum_{j=1}^{J} \left\| \tilde{P}_j^t - P_j^t \right\|_2^2 + \mathcal{L}_{\text{SCL}} \quad (2)$$

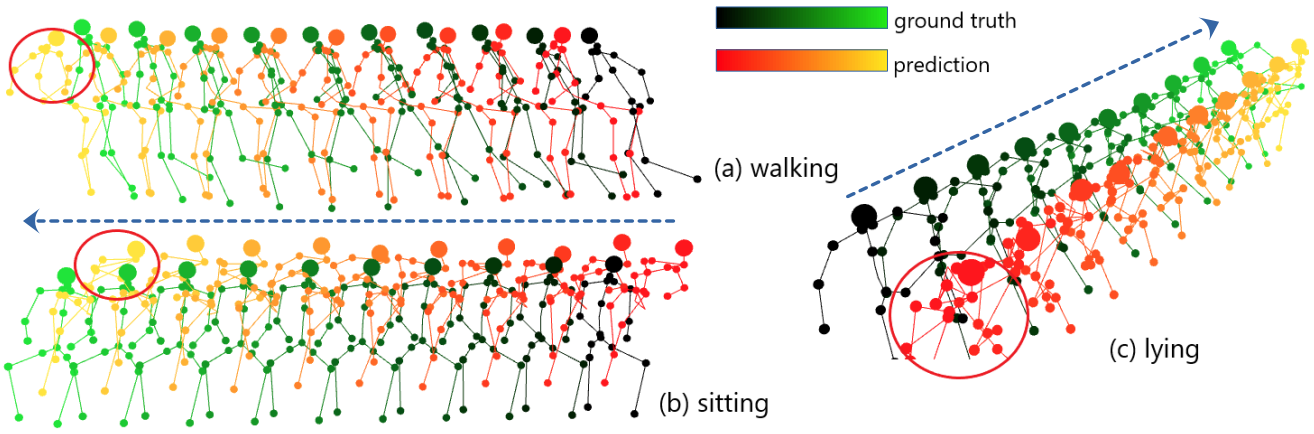

Figure 4: Different cases of deformation of the predicted 3D skeletons. Arrow indicates the time direction. The first two poses (a) and (b) show awkward joint angles between each joint. The case (c) one the right shows failure in both predicted angle and bone length.

5. Experiments

In this section, we describe the datasets used in the training, the training settings, and evaluation metrics. Then we compare with several prior models followed by an ablation study of our model both in quantitatively and qualitatively manners.

5.1. Datasets description

GTA-IM: GTA Indoor Motion Dataset [6] emphasizes on human-scene interactions. The motivation of this dataset is to fix the problem that real datasets of human-scene interaction has which is the noisy 3D human pose annotations and limited long-range human motion. The synthetic data of motions and interactions were collected from 3D video game *Grand Theft Auto V* by controlling characters, cameras, and the physical system. The data set contains 50 human characters acting inside 10 different large indoor scenes. The dataset includes RGBD frames with 1920×1080 resolution, the corresponding ground-truth 3D human pose joints, human skeleton segmentation, and the camera parameters. We split 8 scenes for training and 2 scenes for evaluation following the settings of [6]. We also transfer the 3D path into the camera coordinate frame for both training and evaluation. **PROX:** Proximal Relationships with Object eXclusion (PROX) is a new dataset captured using the Kinect-One sensor by [13]. It contains 12 different 3D scenes with a total of 60 recorded scenarios. Each video is 30 FPS with camera parameters, calibration parameters, and human body segments. 3D skeleton points(25 joints) of the human pose are captured by Kinect-One sensor and 2D keypoints(25 joints) are captured by OpenPose. In our experiment, we split PROX dataset with 52 training sequences and 8 sequences for testing following the settings of [6]. The usage of OpenPose to generate the ground truth makes it less accurate than the GTA-IM dataset. This is because the ground truth will inherit the errors of OpenPose. Also, the PROX dataset was mostly captured in a lab environment making it less diverse in terms of visual features. These flaws of the PROX dataset will impact our results as we will see in the ablation study section. A similar discussion was raised by the authors of [6]. Also, having results on the PROX dataset shows the robustness of our method in the case of noisy estimated 2D poses.

5.2. Evaluation metrics

The main metric for evaluating the 3D path and 3D pose prediction is the Mean Per Joint Position Error (MPJPE) [18]. For a frame $t \in \tilde{T}$ and a skeleton with J joints , MPJPE is computed as:

$$E_{MPJPE}(t) = \frac{1}{J} \sum_{j=1}^{J} \left\| \tilde{P}_j^t - P_j^t \right\|_2 \tag{3}$$

The prior formulation is used to compute the 3D pose error. In case of the 3D path error, a specif joint is chosen such as the center of the skeleton torso. We also use two common metrics that can be found in the literature of pedestrian trajectory prediction [1, 12, 25] to evaluate our performance. First, the Mean Average Displacement Error(ADE) defined in equation 4 which judges the overall performance of both pose and path errors overall the predicted trajectory. Second, the Mean Final Displacement Error(FDE) which judges the performance at the final time step of the trajectory. The FDE is an indicator of the accumulation of the errors in the prediction, in other terms a low FDE means fewer errors were accumulated. The FDE defined in equation 5. Lastly, we define a measure for the stability over the prediction horizon STB_σ Equation 6. It is the average error for both path and pose predictions. If the predictions deviate or accumulate errors over the long-term this metric increases and vice versa.

Though this metric only discusses the divergence over the prediction horizon and it does not indicate the accuracy of the predictions, unlike the ADE and FDE metrics.

$$\text{ADE} = \frac{1}{2}\left(\left[\frac{1}{\tilde{T}}\sum_{t=1}^{\tilde{T}} E_{MPJPE}(t)\right]_{\text{pose}} + \left[\frac{1}{\tilde{T}}\sum_{t=1}^{\tilde{T}} E_{MPJPE}(t)\right]_{\text{path}}\right) \tag{4}$$

$$\text{FDE} = \frac{1}{2}\left(\left[E_{MPJPE}(t=\tilde{T})\right]_{\text{pose}} + \left[E_{MPJPE}(t=\tilde{T})\right]_{\text{path}}\right) \tag{5}$$

$$\text{STB}_\sigma = \sqrt{\frac{1}{2}\left(\text{Var}\left\{E_{MPJPE}(t)|t\in\tilde{T}\right\}_{\text{pose}} + \text{Var}\left\{E_{MPJPE}(t)|t\in\tilde{T}\right\}_{\text{path}}\right)} \tag{6}$$

5.3. Training settings

For all of our experiments, we use SGD optimizer. The initial learning rate for GTA-IM is 0.01 and 0.03 for the PROX dataset. The number of training epochs is 450 and we decrease the learning rate by a factor of 0.2 every 200 epochs. We use a batch size of 128 and 1 second of observation and 2 seconds for predictions following the settings of [6]. We used GTX-1080Ti for training on a 128 GB RAM machine. The need for a large RAM comes when the models are trained using the visual signal. Training each model took between 8 hours and 24 hours depending on the used dataset.

5.4. Comparison with Prior Methods on GTA-IM & PROX Datasets

In this section, we perform quantitative evaluations of our method. The evaluation results of the 3D path and pose predictions on the GTA-IM dataset are shown in Table 1 while the PROX dataset results are in Table 2. Overall, Skeleton-Graph outperforms the prior methods on several metrics. For the FDE we are 105 mm more accurate than GPP-Net [6] on the GTA-IM dataset and 110 mm more accurate than GPP-Net on the PROX dataset. This shows that we did not accumulate error over the long-term prediction, unlike prior methods. The ADE is slightly better than the previous state-of-the-art GPP-Net by 10mm. This means, on average we have a more accurate 3D path and pose predictions. For the divergence in the long-term, the STB_α has a drastic drop in comparison with prior works. We are 88% better on the GTA-IM dataset and 93% better on the PROX dataset. This can be seen in the tables where our 0.5 seconds are close to the 2 seconds prediction in terms of MPJPE which means no divergence happening in the long-term, unlike prior works. We also notice that our model on the 0.5 seconds horizon in GTA-IM does not perform better than the prior methods. The same notice can be seen in the PROX dataset results. We highlight this as a trade-off between accuracy over short-term prediction versus the stability of prediction over the long-term due to not using recurrent approaches. As discussed in the introduction recurrent approaches tend to be accurate in the short term. For example, we notice that prior methods tend to be accurate over the short-term but diverge drastically over the long term, unlike our model. Overall, though our model behaves like this in the short term, the overall average performance is still better than the prior methods by at least 27% and 4% on both the FDE and ADE metrics, respectively.

6. Ablation Study

In this section, we extensively analyze the different behaviors of Skeleton-Graph model in both quantitative and qualitative ways. We start with a quantitative analysis of the different configurations of our model such as the adjacency learning and the skeleton consistency loss. We follow up with a visual analysis of these components and their effect on the results.

6.1. Quantitative analysis of model components

In this section we study the different components of Skeleton-Graph model. The components are: Learning adjacency CNN $+\tilde{\mathcal{A}}$, image embedding $+I_T^{\text{2D}}$, video embedding $+I_{1..T}^{\text{2D}}$ and the two different components of our skeleton consistency loss SCL_{cos}, SCL_{L_2}. Table 3 shows an evaluation of these configuration on both GTA-IM and PROX datasets.

Plain Skeleton-Graph model: We directly train the raw skeleton-graph model without self-learning adjacent matrix, consistency constraints, and visual signal. This raw model alone outperforms the previous work on both the GTA-IM and PROX datasets on both FDE and ADE metrics. This validates that the graph approach we utilize in our model and the full CNN approach can remarkably decrease predictive MPJPE in the pose prediction task. We also notice that the results are divergence-free along the GTA-IM and PROX datasets with STB_σ that is better than prior methods.

Skeleton-Graph with learning adjacency $+\tilde{\mathcal{A}}$: The work of [25] suggested that the kernel function that governs the value of the adjacency matrix influences the results of the trajectory prediction significantly. Overall, the adjacency matrix is important in GCNNs because it governs the interaction between the graph nodes. Instead of searching for a handmade kernel function, we decided to let the model discovers the proper weights for the adjacency matrix. From Table 3 we can see that the usage of the learned adjacency matrix STB_σ improves the performance in comparison with the plain model. This indicates that the model discovered a better interaction between the graph nodes. Figure 6 illustrates a heat map of the learned adjacency STB_σ. We notice an increase in the connections between the two legs joints, the same happens between the two hands joints. This em-

Time step (second)	3D path error (mm) 0.5	1	1.5	2	3D pose error (mm) 0.5	1	1.5	2	FDE	↓ ADE	STB$_\sigma$
TR [36]	277	352	457	603	291	374	489	641	622	436	147
TR [36] + VP [27]	157	240	358	494	174	267	388	526	510	326	150
VP [27] + LTD [38]	124	194	276	367	121	180	249	330	349	230	98
GPP-Net [6]	**104**	**163**	219	297	**91**	**158**	237	328	313	200	93
Skeleton-Graph (ours)	154	**163**	**172**	**186**	198	209	**217**	**230**	**208**	**192**	**11**

Table 1: **Results of the GTA-IM dataset**. Results of 3D path and pose MPJPE error are reported in mm. The lower, the better.

Time step (second)	3D path error (mm) 0.5	1	1.5	2	3D pose error (mm) 0.5	1	1.5	2	FDE	↓ ADE	STB$_\sigma$
TR [36]	487	583	682	783	512	603	698	801	792	644	126
TR [36] + VP [27]	262	358	461	548	297	398	502	590	569	427	126
VP [27] + LTD [38]	194	263	332	394	216	274	335	394	394	300	82
GPP-Net [6]	**189**	**245**	317	389	**190**	**264**	335	406	398	292	90
Skeleton-Graph (ours)	264	269	**272**	**277**	281	287	**291**	**298**	**288**	**280**	**6**

Table 2: **Results of the PROX dataset**. Results of 3D path and pose MPJPE error are reported in mm. The lower, the better.

phasizes that the motion of humans has a strong pattern that is related to the coordination between both the hands and the legs. We also notice that the spine connections are almost the same as the original skeleton joints. This indicates that the spine does not contribute too much to the motion pattern.

Skeleton-Graph +$\tilde{\mathcal{A}}$ + Skeleton Consistency Loss (SCL): Though the model with the learned adjacency matrix performs well, the model will generate many weird human poses without the consistency constraints. We show 3 failure cases in Figure 4 where the angles and distances between joints are out of the normal range of a human body. We add cosine similarity+SCL$_{\cos}$ and joints distance+SCL$_{L_2}$ in the loss function and fine-tune hyper-parameters(optimal weights) to combine the three terms of loss: prediction-target loss, cosine similarity loss, and norm loss. We get improved performance by using these constraints as shown in Table 3 on the GTA-IM dataset. On the other hand, in the PROX dataset, it seems only the+SCL$_{L_2}$ enhances the performance. This is because of the nature of the PROX dataset. The PROX dataset ground truth is machine-generated so it is not accurate when compared with the GTA-IM dataset which the ground truth is obtained from the game directly. This is behavior was seen before in the work of [6]. However, adding consistency loss does increase the training time in comparison with the plain model. Yet, it results in more accurate FDE and ADE results and more natural-looking skeletons as we will see in the qualitative study section.

Skeleton-Graph +$\tilde{\mathcal{A}}$+SCL+ visual signal: We start with the GTA-IM dataset. The prior work of [6] shown that using the visual signal of the last observed frame enhances the performance. From Table 3 we notice that using the last frame $+I_T^{2\text{D}}$ did enhance the short-term 3D path error in comparison with the previous components on the GTA-IM dataset. Yet, it did not enhance the short-term 3D pose error. When we used the full sequence of observed images $+I_{0..T}^{2\text{D}}$ on the GTA-IM dataset, the 3D pose error was the lowest among all the modes of configurations. This indicates that it helped predicting the 3D poses. Yet, for the 3D path, it had the highest error among all the configurations. This shows a trade-off between the path and poses objectives influenced by the presence of the visual signal. Overall, using the vision signal resulted in a performance that is similar to the usage of the consistency objective on the ADE and FDE metrics with a noticeable inner performance enhancement in the short term of both path and pose predictions. For the PROX dataset using the vision signal either the last image, $+I_T^{2\text{D}}$ or the full sequence $+I_{1..T}^{2\text{D}}$ resulted in a divergence of the results. This aligns with the findings of [6] over the PROX dataset. In addition, the PROX were captured in an empty lab environment, and thus it no as rich as the GTA-IM dataset. This made the dataset to be less diverse in terms of the background and the camera poses. So inherently the vision signal becomes less useful leading to ambiguity in the predictions as seen in Table 3.

6.2. Qualitative study of the models' components

To understand the effect of each configuration in our model, we visualize the predicted 3D human poses per each configuration as shown in Figure 5. Starting from the standard model with self-learning adjacent matrix alone $+\tilde{\mathcal{A}}$. The entire predicted pose looks natural in some areas and unnatural in other areas. The predicted right leg stretches out naturally and the left leg bends just like the target. However, the left hand in the pose is too close to the head and looks weird since there is no norm loss SCL$_{L2}$ in the objective function. After we add norm loss(seen on the top right), the prediction has a reasonable left hand but the right leg joints penetrate the ground which is not possible in reality. Once the cosine loss being added SCL$_{cos}$ to the objective, the penetration problem is gone but the distances between joints

Dataset	Skeleton-Graph configuration					3D path error (mm)				3D pose error (mm)				↓		
	$+\tilde{\mathcal{A}}$	+SCL$_{\cos}$	+SCL$_{L_2}$	$+I_T^{2D}$	$+I_{0..T}^{2D}$	0.5	1	1.5	2	0.5	1	1.5	2	**FDE**	**ADE**	**STB$_\sigma$**
GTA-IM						150	165	175	191	198	211	220	234	213	193	16
	✓					159	169	178	192	196	208	216	229	211	193	14
	✓	✓				161	169	176	187	199	209	217	230	208	193	10
	✓		✓			159	169	177	191	**191**	204	212	225	208	191	14
	✓	✓	✓			159	168	177	190	193	205	213	226	208	191	11
	✓	✓	✓	✓		**154**	**163**	**172**	**186**	198	209	217	230	208	191	11
	✓	✓	✓		✓	165	174	181	193	190	**202**	**211**	**224**	208	192	11
PROX						340	348	353	360	369	375	379	386	373	364	7
	✓					309	314	317	323	335	339	342	347	335	328	5
	✓	✓				280	287	290	296	297	303	307	314	305	296	5
	✓		✓			**264**	**269**	**272**	**277**	**281**	**287**	**291**	**298**	288	280	6
	✓	✓	✓			293	299	302	308	292	298	302	308	308	300	5
	✓	✓	✓	✓		353	353	355	360	358	362	365	370	365	359	3
	✓	✓	✓		✓	283	290	294	301	308	314	319	325	313	304	6

Table 3: **Skeleton-Graph ablation**. $+\tilde{\mathcal{A}}$ is the model with learned adjacency. SCL$_{\cos}$ and +SCL$_{L_2}$ indicates the usage of the SCL components. The usage of last image is $+I_T^{2D}$, $+I_{1..T}^{2D}$ indicates the usage of whole observed sequence. All results are in mm, the lower the better. The (0.5, 1, 1.5, 2) are time steps in seconds. Bolded numbers are the best in each column.

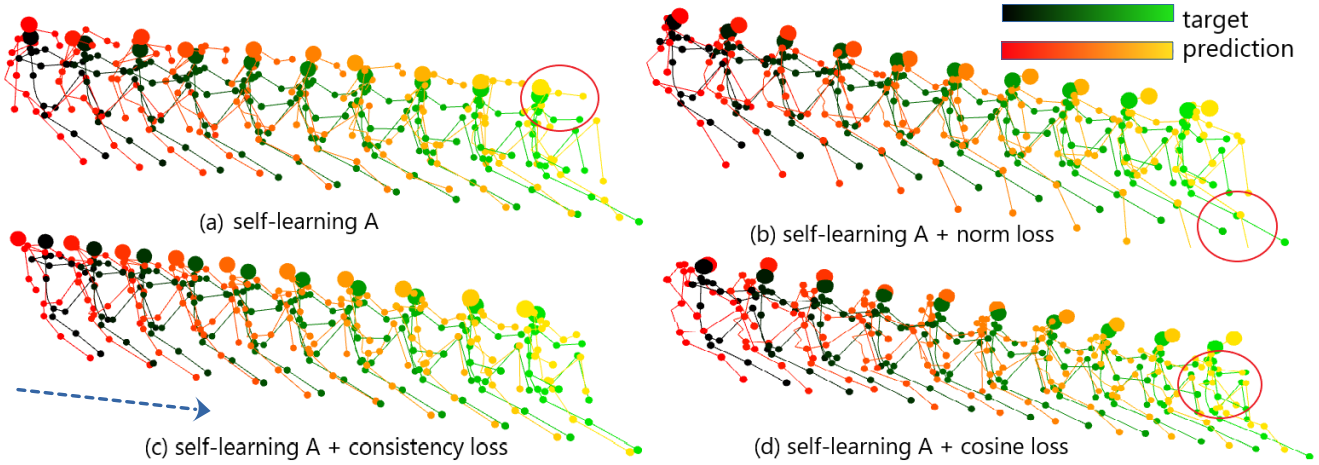

Figure 5: Effect of different configurations of Skeleton-Graph model on the 3D pose predictions. Arrow indicates the time direction. Top left:Skeleton-Graph model with learned adjacency $+\tilde{\mathcal{A}}$; Bottom left: Skeleton-Graph model $+\tilde{\mathcal{A}}$ + weighted $\mathcal{L}_{\text{SCL}}$; Top right: Skeleton-Graph model $+\tilde{\mathcal{A}}$ +SCL$_{L2}$ only; Bottom right: Skeleton-Graph model $+\tilde{\mathcal{A}}$ +SCL$_{cos}$.

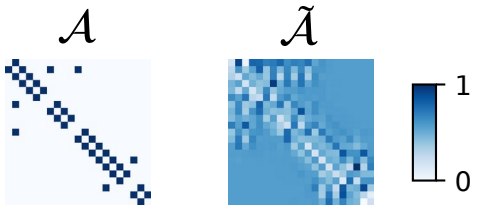

Figure 6: Heat map of the learned adjacency matrix $\tilde{\mathcal{A}}$ versus the original adjacency that represents the connection between the skeleton joints.

become unnatural especially in the first five frames(seen on the bottom right). When both SCL$_{L2}$ and SCL$_{cos}$ are used in training our model the resulted skeletons look very natural as shown in the bottom left in Figure 5. More qualitative cases are in the supplementary materials.

7. Conclusion

We showed that the usage of graph CNNs with self-learning adjacency matrix and formulating the problem as a spatio-temporal graph is suitable for the problem of 3D skeleton motion predictions. We achieved state-of-the-art results on well-known benchmarks. The design of our model results in divergence-free predictions in the long term, unlike prior works. This was shown using the introduced STB$_\sigma$ metric. The deformation in prediction was solved using a skeleton consistency loss. The integration of the vision signal improved the results on the GTA-IM dataset. In the future, we would like to target the short-term prediction accuracy issue and investigate different methods for integrating the visual signals.

Acknowledgement: We thank the reviewers for their feedback. This research was supported by NSF (National Science Foundation) CPS No.1739964 and the CAMMSE UTC (US DOT).

References

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 961–971, 2016. 5

[2] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018. 3

[3] Artur Bąk, Marek Kulbacki, Jakub Segen, Dawid Świątkowski, and Kamil Wereszczyński. Recent developments on 2d pose estimation from monocular images. In *Asian Conference on Intelligent Information and Database Systems*, pages 437–446. Springer, 2016. 1, 4

[4] Ernesto Brau and Hao Jiang. 3d human pose estimation via deep learning from 2d annotations. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 582–591. IEEE, 2016. 4

[5] Yujun Cai, Liuhao Ge, Jun Liu, Jianfei Cai, Tat-Jen Cham, Junsong Yuan, and Nadia Magnenat Thalmann. Exploiting spatial-temporal relationships for 3d pose estimation via graph convolutional networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2272–2281, 2019. 2

[6] Zhe Cao, Hang Gao, Karttikeya Mangalam, Qi-Zhi Cai, Minh Vo, and Jitendra Malik. Long-term human motion prediction with scene context. In *European Conference on Computer Vision*, pages 387–404. Springer, 2020. 1, 2, 3, 5, 6, 7

[7] Ching-Hang Chen and Deva Ramanan. 3d human pose estimation= 2d pose estimation+ matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7035–7043, 2017. 2

[8] Rishabh Dabral, Anurag Mundhada, Uday Kusupati, Safeer Afaque, Abhishek Sharma, and Arjun Jain. Learning 3d human pose from structure and motion. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 668–683, 2018. 4

[9] Wenhao Ding, Shuaijun Li, Guilin Zhang, Xiangyu Lei, and Huihuan Qian. Vehicle pose and shape estimation through multiple monocular vision. In *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 709–715. IEEE, 2018. 1

[10] Renshu Gu, Gaoang Wang, and Jenq-Neng Hwang. Efficient multi-person hierarchical 3d pose estimation for autonomous driving. In *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 163–168. IEEE, 2019. 1

[11] Renshu Gu, Gaoang Wang, and Jenq-neng Hwang. Efficient multi-person hierarchical 3d pose estimation for autonomous driving. In *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 163–168, 2019. 1

[12] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2255–2264, 2018. 5

[13] Mohamed Hassan, Vasileios Choutas, Dimitrios Tzionas, and Michael J Black. Resolving 3d human pose ambiguities with 3d scene constraints. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2282–2292, 2019. 2, 5

[14] Chong Huang, Fei Gao, Jie Pan, Zhenyu Yang, Weihao Qiu, Peng Chen, Xin Yang, Shaojie Shen, and Kwang-Ting Cheng. Act: An autonomous drone cinematography system for action scenes. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7039–7046. IEEE, 2018. 1

[15] Chong Huang, Zhenyu Yang, Yan Kong, Peng Chen, Xin Yang, and Kwang-Ting Tim Cheng. Learning to capture a film-look video with a camera drone. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1871–1877. IEEE, 2019. 1

[16] Yingfan Huang, Huikun Bi, Zhaoxin Li, Tianlu Mao, and Zhaoqi Wang. Stgat: Modeling spatial-temporal interactions for human trajectory prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6272–6281, 2019. 2

[17] Zhen Huang, Xu Shen, Xinmei Tian, Houqiang Li, Jianqiang Huang, and Xian-Sheng Hua. Spatio-temporal inception graph convolutional networks for skeleton-based action recognition. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2122–2130, 2020. 2

[18] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, 2014. 5

[19] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 2

[20] Guohao Li, Matthias Muller, Ali Thabet, and Bernard Ghanem. Deepgcns: Can gcns go as deep as cnns? In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9267–9276, 2019. 11

[21] 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. 2

[22] Sijin Li and Antoni B Chan. 3d human pose estimation from monocular images with deep convolutional neural network. In *Asian Conference on Computer Vision*, pages 332–347. Springer, 2014. 2

[23] Srikanth Malla, Chiho Choi, and Behzad Dariush. Socialstage: Spatio-temporal multi-modal future trajectory forecast. *arXiv preprint arXiv:2011.04853*, 2020. 3

[24] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3d human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2640–2649, 2017. 2

[25] Abduallah Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian Claudel. Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Com-*

puter Vision and Pattern Recognition, pages 14424–14432, 2020. 2, 3, 5, 6

[26] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016. 2

[27] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3d human pose estimation in video with temporal convolutions and semi-supervised training. In *CVPR*, 2019. 1, 2, 7

[28] Sebastian Pohl, Armin Becher, Thomas Grauschopf, and Cristian Axenie. Neural network 3d body pose tracking and prediction for motion-to-photon latency compensation in distributed virtual reality. In *International Conference on Artificial Neural Networks*, pages 429–442. Springer, 2019. 1

[29] Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Reconstructing 3d human pose from 2d image landmarks. In *European conference on computer vision*, pages 573–586. Springer, 2012. 4

[30] CH Rodriguez-Garavito, Guillermo Camacho-Munoz, David Álvarez-Martínez, Karol Viviana Cardenas, David Mateo Rojas, and Andrés Grimaldos. 3d object pose estimation for robotic packing applications. In *Workshop on Engineering Applications*, pages 453–463. Springer, 2018. 1

[31] Grégory Rogez and Cordelia Schmid. Mocap-guided data augmentation for 3d pose estimation in the wild. *arXiv preprint arXiv:1607.02046*, 2016. 1

[32] Nikolaos Sarafianos, Bogdan Boteanu, Bogdan Ionescu, and Ioannis A Kakadiaris. 3d human pose estimation: A review of the literature and analysis of covariates. *Computer Vision and Image Understanding*, 152:1–20, 2016. 1

[33] Mingyi Shi, Kfir Aberman, Andreas Aristidou, Taku Komura, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. Motionet: 3d human motion reconstruction from monocular video with skeleton consistency. *ACM Transactions on Graphics (TOG)*, 40(1):1–15, 2020. 4

[34] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 648–656, 2015. 2

[35] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014. 2

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017. 1, 7

[37] Chengxin Wang, Shaofeng Cai, and Gary Tan. Graphtcn: Spatio-temporal interaction modeling for human trajectory prediction. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3450–3459, 2021. 2, 3

[38] Mao Wei, Liu Miaomiao, Salzemann Mathieu, and Li Hongdong. Learning trajectory dependencies for human motion prediction. In *ICCV*, 2019. 1, 2, 3, 7

[39] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *Proceedings of the European conference on computer vision (ECCV)*, pages 466–481, 2018. 2

[40] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Thirty-second AAAI conference on artificial intelligence*, 2018. 2, 3

[41] Dapeng Zhao and Jean Oh. Noticing motion patterns: Temporal cnn with a novel convolution operator for human trajectory prediction. *IEEE Robotics and Automation Letters*, 2020. 3

[42] Xiaowei Zhou, Sikang Liu, Georgios Pavlakos, Vijay Kumar, and Kostas Daniilidis. Human motion capture using a drone. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 2027–2033. IEEE, 2018. 1

A. Failure Cases

Figure 4 illustrates several failure cases by our model. In case (1) we notice a skeleton sitting but our model was not able to capture the torso crouch to represent this mode. Case (2) the predicted skeleton trajectory overshoots, in other terms the model predicted too much of a momentum from the history of the observations. Case (3) and (4) are complex situations where the target skeleton changes from sitting to standing, we notice our model was not able to capture this trend. This is probably because of the lack of history that represents this trend in the 2D observations. Case (5) and (6) is a skeleton sitting or relaxing on an object, though our prediction are close, they look abnormal. This happens because in these two cases the joints angles vary a lot in which the predictions become harder. We believe in the future adding more dynamics oriented constraints can enhance these prediction errors.

B. Skeleton-Graph Model Configuration

We found out that the number of TXCNN and SPGCNN layers affects the model performance significantly. Table 5 we show the effect with the number of TXCNN and SPGCNN on the model performance. We used residual connection while going deeper using this layers. First, we notice that going deeper with the number of SPGCNN results in a significant performance drop on both path and pose estimation. This is the same behavior noted in [20]. Going deeper with the number of TXCNNs beyond 11 layers resulted in a drop of the performance. We notice that the performance for both 9 and 11 TXCNN layer are close to each other. Also, the result of the ablation is the same on both PROX and GTA-IM datasets. This means that our model is expected to behave in the same way on different datasets, a kind of agnostically to the dataset.

C. Model Architecture

Table 6 shows the inner details of our model. The structure of each component in terms of the parameters of the CNN layers, the usage of batch norm and the location of the activation functions are all shown in this table.

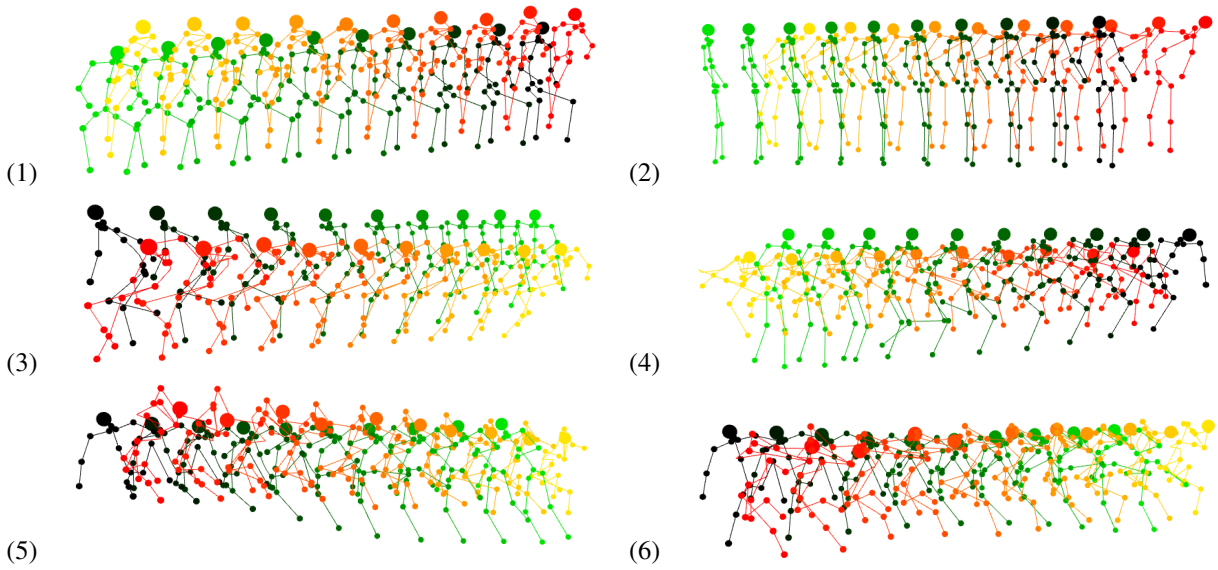

Table 4: Several failure cases predicted by Skeleton-Graph . The green skeletons represents ground-truth and red skeleton represents prediction. The deeper color the skeleton is, the earlier moment it is corresponding to.

Dataset		1	3	5	7	9	11
GTA-IM	1	320/362	267/290	211/247	221/250	213/237	230/245
	3	262/296	252/278	230/251	198/246	212/239	210/225
	5	272/298	273/277	234/257	206/234	203/241	**202/231**
	7	271/295	264/288	233/263	208/250	233/260	233/260
PROX	1	413/455	361/373	340/319	339/361	367/314	336/347
	3	462/454	433/506	362/397	403/372	346/345	376/367
	5	499/416	397/441	426/414	439/468	741/443	**351/274**
	7	582/679	455/420	400/408	420/426	409/484	371/351

Table 5: The effect of the choice of the number of TXCNN and SPGCNN on prediction results. The row is for the number of TXCNN layer while the column is for the number of SPGCNN layers. All readings are in mm, the lower the better. The numbers are Path/Pose errors. The error is over 2 seconds prediction horizon.

Component	Layer name / description	Layer strucutre
Spatio-Temporal Graph CNN	Input	CNN(2,3,k=3,p=1)
	Graph	GraphCNN(3,3)
	Learn Adjacency	CNN(3,3) BN,PReLU,CNN(2,3,k=3,p=1),BN
	Temporal CNN	CNN(3,3) BN,PReLU,CNN(2,3,k=3,p=1),BN
Vision Features Extractor		CNN(3,6,k=3,p=1), BN, PReLU, CNN(6,9,k=3,p=1,s=2), BN, PReLU, CNN(9,12,k=3,p=1,s=2), BN, PReLU, CNN(12,15,k=3,p=1,s=2), BN, PReLU, CNN(15,18,k=3,p=1,s=2), BN, PReLU, CNN(18,21,k=3,p=1,s=2), BN, PReLU
Concatenate		Concatenate spatio-temporal graph CNN output with the vision features extractor output
Time Extrapolator CNN	Input	CNN(T+C,$\tilde{T}$,3,p=1), BN, PReLU
	Middle	CNN($\tilde{T}$,$\tilde{T}$,k=3,p=1), BN, PReLU + residual
	Output	CNN($\tilde{T}$,$\tilde{T}$,k=3,p=1),

Table 6: Skeleton-Graph architecture description. CNN = Convolutional Neural Layer, k= kernel, p= padding, s= stride, BN= Batch Normalization, PReLU = Parametric ReLU activation function, T= observed time steps, C= vision signal features channels and $\tilde{T}$= predicted time steps.