Human Motion prediction with Scene Context

Benjamin Dayan
ETH Zurich
bdayan@student.ethz.ch

Di Zhuang
ETH Zurich
dzhuang@student.ethz.ch

Xiyi Chen
ETH Zurich
xiychen@student.ethz.ch

Abstract

Human motion prediction is a popular yet challenging task in computer vision. Existing methods either ignore scene context or focus on partial scene context of certain key points. In this work, we aim to incorporate human-environment interactions into motion prediction via proximity data between joints and the surroundings. We develop a framework to compute proximity maps and perform data pipelining to sequentialise normalized 3D joint locations from the PROX dataset [7]. We propose RNN and Transformer based models to perform human motion prediction directly from 3D joint locations, and discuss a few alternative model designs. Qualitative and quantitative evaluations performed on a validation set from PROX dataset show that our models can predict natural human movement trajectories with correct shapes with 5 input frames and 10 prediction frames, but the overall ability to match the ground-truth trajectory is limited. We provide a detailed discussion on the explanation of limitations and future works.

1. Introduction

Human motion prediction is one of the most popular tasks of computer vision in recent years, as it is crucial for applications such as self-driving cars, autonomous mobile robots, or virtual reality, where motions of virtual humans need to be planned realistically.

Existing methods of human motion prediction either ignore scene context or focus on partial scene context of certain key points, while, in reality, the motions of humans are often influenced or in reaction to their physical surroundings. Therefore, to achieve realistic prediction of human motions, it is important to consider the human-environment interactions in the scene context.

In visual computing, a common representation of scene context is RGBD video, which, for each frame, consists of an RGB image and a corresponding depth image. Similar as [16] , we can encode human-environment interactions via proximity by computing the distance between each body part and its closest point in the scene. It is expected that this additional information could improve the performance of existing human motion prediction models.

This project aims to learn a temporal model to predict future human motions given past motions and proximity maps computed from the RGB-D frames of the scene. In particular, we can use a pre-trained CNN (e.g. ResNet [8]) to extract feature vectors from the proximity maps, as they're images. After that, for each frame, we can append the 3D joint locations of the person to the proximity feature vector, and build a network to regress the future 3D joint locations of the person from the past proximity feature vectors and 3D joint locations. Due to time constraints, we are only able to develop a framework to produce the proximity maps but do not incorporate them into training our temporal models.

In summary, our key contributions are the following:

- We perform data preprocessing and pipelining on PROX dataset [7] to extract and sequentialize 3D joint locations.
- We develop a framework to compute proximity maps from depth maps for PROX.
- We train and evaluate two models, an RNN-based model and a Transformer-based model, to predict future human motion directly from 3D joint locations, and compare their performances.

2. Related Work

There has been plenty of interest in human motion prediction in real-world scene context. Most of the existing approaches involve an encoding of the human-scene interactions and a temporal model that predicts future human motions based on past human motions and encoding of human-scene interactions. Therefore, we mainly focus on understanding of human-scene interactions and human motion prediction for related work.

2.1. Human-scene Interactions Understanding

Machine perception of human-environment interactions has been extensively studied. [4] proposes a pairwise body-

part attention model that detects interactions between objects and crucial parts of human body. [12] develops a model that learns 2D-3D joint representation for human-object interactions. [6] builds a human-centric paradigm for scene understanding, which allows prediction of potential human pose and joint locations in a given indoor scene. [11] presents a model that learns to insert an object instance into an image using a semantic label map. [16] generates realistic human-object interactions by explicitly modeling the proximity relationship between human and objects.
In our project, we use a proximity map that represents human-scene interactions with distances between each part of the human body and the point cloud of the scene. This is a human-centric representation for scene context that is simple to compute given a depth image and knowledge of human joint locations, and has a convenient structure as an image, suitable for use with CNNs.

2.2. Human Motion Prediction

Human motion prediction has been a long standing problem. Early work models the motion of pedestrians as if they would be subject to 'social forces', and predict future human motions using hand-crafted functions [9]. Recent work propose temporal models, using a recurrent neural network (RNN), such as an LSTM model in [1], that predicts future human trajectories. As the importance of scene context and human-scene interactions has been emphasized, works have been done to incorporate this information into prediction tasks. [3] proposes a framework that predicts future human poses and locations given a single scene image and past 2D human poses by factorizing the problem into motion goal prediction, path planning and 3D pose generation, where scene context is represented by a single RGB image.
We propose to predict future human motions given past motions and proximity maps computed from the RGB-D frames of the scene. In comparison to a single scene image, the information encoded in a proximity map is expected to be more human-centric, and thus more suitable for tasks of human motion prediction.

3. Method

3.1. Sequence Prediction

We follow Martinez et al.'s [13] approach and perform human joint location prediction in a sequence-to-sequence (seq2seq) manner using two approaches: RNN and transformer.

We initially incorrectly try to predict 30 future frames from 15 input frames, at 3 FPS (predict 10 seconds into the future from 5 seconds of past observation). This is overly ambitious which crosses the line between deterministic and stochastic ambition. The longer time scale of 15 total seconds becomes more a motion generation task. This would require a different training structure - common approaches involve GANs (reinforcing realistic motion by a (simultaneously trained) real vs fake motion discriminator) and VAE's (encoder decoder model where motion is generated from the encoded latent vector - reinforce reproduction of original motion (decoding of latent original sequence), while incentivising a good distribution in latent vector space).

We later turn switch to predicting 10 frames from 5 at 5 FPS (2 seconds future from 1 second past). In this shorter time period there's little room for a human to drastically change their chain of motion, so a decent prediction can be achieved.

3.2. Data Preparation

3.2.1 PROX Dataset

We use the PROX dataset [7] which contains 100K frames at 30 FPS; it's roughly 50 minutes of footage of human motion in indoor scenes. PROX has 12 different 3D indoor environments with 20 subjects moving and interacting inside - we originally chose it for the ability to compute proximity maps from depth camera footage with fitted 3D human joint location, and because the human motion within is very scene interactive - sitting/lying/rolling around on beds/sofas etc.

Due to the small number of sequences we can generate from PROX dataset, we allow a significant overlap between time sequences. We eventually settle on a frame jump between the starting frames of consecutive time sequences as small as 3 frames (at 30FPS this is just 0.1 seconds), giving around 20,000 training sequences for the 5 input frames 10 output 5FPS = 3 second training sequences.

3.2.2 3D Joint Location Generation

A SMPL-X [14] fitting with shape, pose and global translation/orientation parameters is provided for each video frame in PROX with respect to the color coordinate system.

We create a SMPL-X body model and use it to produce 3D joints J as :

$$J(\beta) = \mathcal{J}\left(\bar{T} + B_{\mathcal{S}}\left(\beta; S\right)\right), \tag{1}$$

where $\bar{T}$ represents the template mesh (vertices in the rest pose); β represents shape parameters; $\mathcal{J}$ represents a sparse linear regressor that regresses 3D joint locations from mesh vertices; $B_S\left(\beta; S\right) = \Sigma_{n=1}^{|\beta|}\beta_n \mathcal{S}_n$ represents the shape blend shape function with orthonormal principle components of vertex displacements capturing shape variations due to different person identity, $\mathcal{S}$.

We rotate the k-th joints in rest pose θ^*, $\bar{j}_k$, according to the pose vector θ, and then extract the first 25 3D joints that represent the major keypoints of the human skeleton, to be

our joints to predict, $\bar{j'}_k$:

$$\bar{j'}_k = G'_k(\theta, J)\bar{j}_k, \tag{2}$$

where $G'_k(\theta, J) = G_k(\theta, J)G_k(\theta^*, J)^{-1}$ is the transformation of that joint out of rest pose into our specific pose θ, via the kinematic chain.

To ensure consistency of joint locations across all scenes, we rotate and translate them to the world coordinates using the scene camera rotation vector $R \in \mathbb{R}^{3x3}$ and translation vector $t \in \mathbb{R}^3$. Afterwards, for each sequence of input and future frames, we translate all frames so that the pelvis joint (the first of our 25 joints) of the first frame, p_1, is at the origin. A 3D joint location vector $\bar{j'}_k$ is then normalized to $\hat{j'}_k$ by:

$$\hat{j'}_k = R\bar{j'}_k + t - (Rp_1 + t) = R(\bar{j'}_k - p_1) \tag{3}$$

We should have, but did not, additionally rotated each skeleton about the axis pointing straight along gravity's vector so the first frame skeleton "faces straight north". However, in the case where we would additionally employ a sequence of proximity maps as auxiliary model input, not rotation normalising is the best - we produce the proximity map (see 3.2.3) from the depth camera which is static so has a different view point on the human depending on their rotation. Rotation normalising joints while being unable to rotation normalise the proximity maps would lead to a lopsided dataset.

There are occasional missing or corrupted frame files, and also some skeletons with wildly misplaced locations. For loading our time sequences if one frame has a problematic file, we drop the whole sequence. A better approach could be to linearly interpolate if it is just a single missing skeleton.

3.2.3 Proximity Maps Generation

Each PROX video frame provides 512x424 depth images and 1920x1080 color images along with scene meshes and camera intrinsics/extrinsics, so we can transform points between the depth, color and world coordinate systems. To enhance motion prediction with scene contexts, we calculate proximity maps based on these provided images. Proximity maps are similar to depth images - $M \times N$ 2D arrays where pixel value at point (i, j), $z(x_i, y_j)$, is calculated as:

$$z(x_i, y_j) = d(\vec{p}_{ij}, 0) = ||\vec{p}_{ij}||;, \tag{4}$$

where $i \in |M|$, $j \in |N|$, and $\vec{p}_{ij}$ is the 3D location of one point (i, j) out of a point cloud of $M \times N$ points, with position recorded in the coordinate system of the recording depth camera (position at origin 0). A human proximity map instead sets:

$$z(x_i, y_j) = \min_{\vec{h} \in H} ||\vec{p}_{ij} - \vec{h}||^2, \tag{5}$$

where $H = \{\vec{h}_i : i = 1, ..., 25\}$ is the set of 25 human 3D joints in the same coordinate frame. Because many of the pixel/points of the original depth map are actually collisions with the human, we set these values to 0 (via a 2D human mask on the image). Far away pixels for which a point is not recorded are set to a fixed large value.

A proximity map is easy to calculate given a depth camera and base human joint locations (see figure 1 as an example). As an image, it can be encoded effectively with a CNN. It contains information of nearby objects in the scene to the human ("scene context") - think floor beneath their feet, chair on which they're sitting, wall on which they're leaning, table etc. These are important information that could inform motion prediction - there might be hard constraints such as "they cannot move anymore to the left as then they'd be penetrating into the table", and guiding constraints like "their arm is close to the table, maybe they're moving to lay it down on the table". This is similar to the original goal of the PROX dataset - as a single image human mesh estimation by enforcing no interpenetration between human and objects, but encouraging close proximity of certain body parts with scene - bum, upper back legs and so on.

An idea to enhance this would be to take inverse human joint distance instead - close by scene points are much more important. We could keep the setting of human mask to 0.0 so it contrasts well with the nearby scene pixels, or perhaps introduce the human mask in as a separate color channel, see figure 2 as an example.

A CNN encoded version of a proximity map could be fed into our RNN/Transformer as an additional input (concatenated with the encoded human joint locations). There would need to be proper normalisation - i.e. joint locations should be transformed into the coordinate system of the depth camera (orientation-wise consistency - we can still translate the joints to have pelvis at origin).

3.3. RNN

Our RNN design is based on Martinez et al. paper [13]. The RNN is a stack of GRU layers which sequentially takes in an encoded input vector (human joint locations - (25,3), fed through some FC layers) for each time step. For each of these inputs it updates its current time step internal state (representing its derived knowledge of to date motion of the human), and outputs a timestep t+1 vector which can be decoded (more FC layers) into a new human joint vector.

This differs from the normal "translation" style encoder-decoder seq2seq structure in that the the sequence encoder and decoder are identical with shared weights, and indeed that the encoder already outputs its own joint predictions, predicting the next frame of the input sequence. This was a design choice for simplicity, and even allows additional loss calculations on the input sequence predictions, not just the

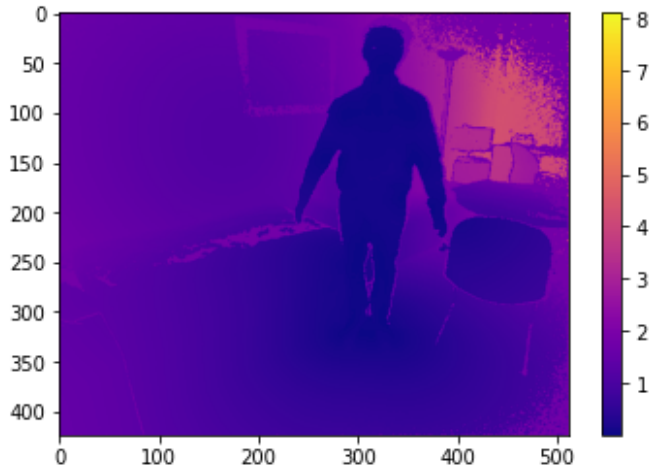

Figure 1. Example of a proximity map from PROX dataset [7], distance units in metres.

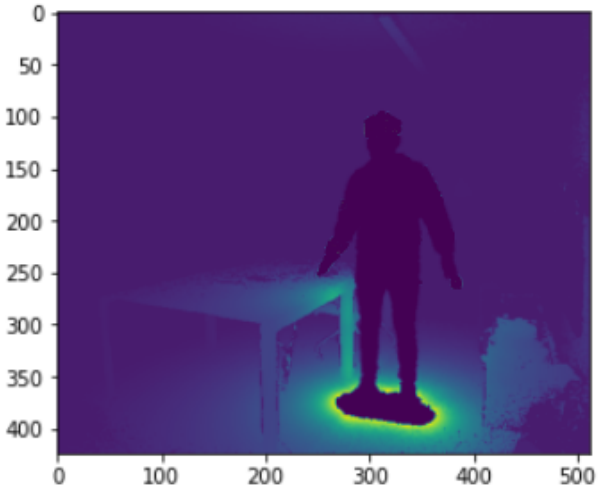

Figure 2. Inverse proximity map with manually adjusted human masking - otherwise feet would be distinct. From PROX dataset [7]

future sequence predictions.

So with an input sequence of $t = 1, ..., n$ frames, frames $t = 1, ..., n-1$ are fed into the encoder RNN, producing already predictions for $t = 2, ..., n$. Then the final input frame n is fed into the (identical) decoder RNN along with the output internal state from the encoder. The decoder predicts the next future frame, which is fed back in the desired number of times for $t = n+1, ..., n+k$ future frame predictions. See Figure 3 for a schematic.

The whole predictor has a residual layer from direct human joint input to output which makes sure that the RNN can focus on figuring out the joint velocities. Something we could have tried is putting in the timestep joint position delta which could be helpful too.

3.3.1 Alternative RNN designs

We give these alternative designs to put ours into context.

- For producing the predicted future frame $t+j$, instead of feeding previous prediction $t+j-1$ into the decoder, the ground truth frame $t+j-1$ can be used. This approach is used by e.g. Fragkiadaki et al. [5], adding noise to the ground truth frame to simulate straying of predicted joint locations off the true path to aid self correcting of errors in testing/inference where you really won't have ground truth future frames. We additionally tried this approach (without the noise) to match the Transformer training procedure.

- Despite having the shared sequence encoder-decoder architecture, Martinez et al. [13] doesn't actually train with loss from predicted future input frames (from the encoder stage). We added this in on advise from Siwei.

- Having a decoupled (weights or additionally structure) encoder-decoder architecture

3.4. Transformer

We also try a transformer model [15] to predict our joint sequences. We first apply embedding with 2 fully-connected layers on the 75 dimensional joints vector to $d_{model} = 200$ dimensional latent vectors. In order to make use of the order of the frames, we inject information about relative/absolute position of the joints in the sequence by applying positional encoding PE for each of the t-th frame with i-th dimension as:

$$PE_{(t,2i)} = sin\left(t/10000^{2i/d_{model}}\right)$$
$$PE_{(t,2i+1)} = cos\left(t/10000^{2i/d_{model}}\right) \quad (6)$$

We then input the position encoded latent vectors of past and future frames into a transformer with 8 heads, 6 encoder layers, 6 decoder layers and a dropout rate of 0.1. Finally, we apply 2 fully-connected layers on the predicted 200 dimensional latent vector for the future sequence and compress it back to 75 dimensions as our predicted joints. We use a learning rate of $1e-4$ to train the model for 200 epochs.

4. Experiments

4.1. RNN/Transformer Experiments

We lacked a consistent experiment protocol which was unhelpful. Initially all experiments were done with 15 input frames, 30 future frames at 3 FPS, only later did we

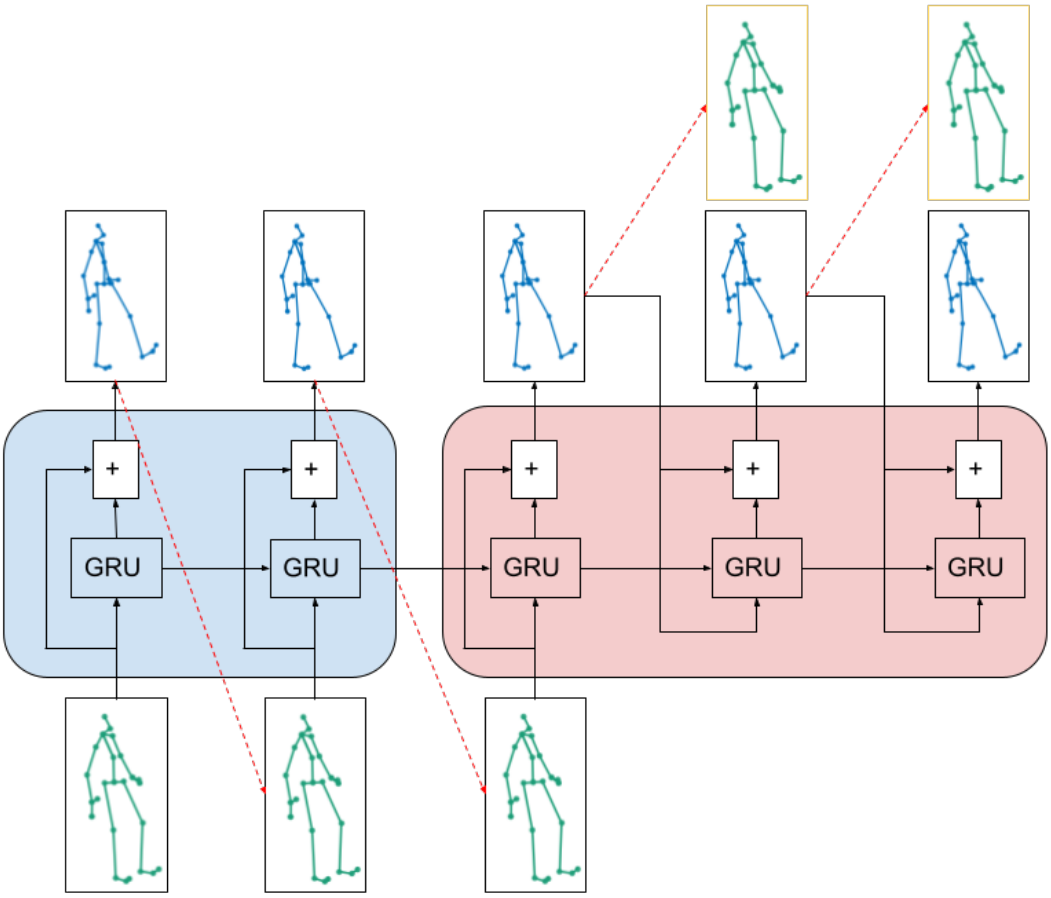

Figure 3. red dashed lines denote loss calculation - mean euclidean distance across all 25 joints. GRU box is the RNN + pre and post FC (input/output encoding/decoding). The left blue oblong is the time sequence Encoder portion, the right red oblong is the time sequence decoder. Here there are 3 input frames and 2 future prediction frames

switch to 5 input frames, 10 future frames at 5 FPS (easier and more deterministic prediction). Also the Transformer architecture differs from the final RNN design in that the transformer always uses ground truth in input and future frames fed into to be encoded, and so would generally have much lower training losses (effectively only having to predict one frame into the future not multiple), as well as for some reason not utilising residual connection to connect input joints to output joints. We can however compare their validation losses. Also useful are qualitative comparisons looking at joint prediction videos.

4.1.1 RNN hyperparameters

We trained for 4 or 8 hours on a NVIDIA RTX 2080 Ti GPU, but most often optimal validation loss was achieved after about an hour of training. Depending on the configuration this gave about 50-150 full epochs of training.

We tried about $10^{-3} : 10^{-5}$ learning rate.

The RNN stack is layers of GRU cells - depth allowing more representational complexity. This gave us two main model structure hyperparameters: number of layers in RNN, and hidden state size for the layers (convention is to have the same size of hidden state vector for each layer). We didn't actually play with encoding and decoding FC hyperparams, and mainly varied `n_layers=2,3,4` and `hidden_size=256,512`

4.1.2 Qualitative Results

1. *15 frames in 30 out 10 second prediction*: skeleton floats off in some direction while slowly contorting unnaturally

2. *5 frames in 10 out 2 second prediction*: skeleton moves

in the most likely direction according to input motion. The predicted motions still could not follow the trajectory of the ground truths well, but the predicted trajectories are natural and the human shapes are maintained well.

One notable failure of prediction is in human walking - predicted human legs more float in the forward direction rather than pacing. Walking should be textbook predictable human motion - we believe our failure may be due to the PROX dataset having uniquely little "proper" straight line walking compared to other sliding/sitting/shuffling motions, and perhaps also the short sequence lengths of training not providing enough context for walking.

To qualitatively evaluate prediction results on the train/validation set, we produce 2D and 3D videos of human meshes/skeleton joints moving about the scene. We have to convert between the depth, color and world (scene mesh) coordinate systems.

See `https://github.com/BenjiDayan/prox` for visualization of qualitative evaluation on our validation set.

4.1.3 Quantitative Results

	15 ->30 frames	5 ->10 frames
RNN similar as [13]	0.1701	**0.0474**
Transformer [15]	0.0878	0.0557
Replication Loss	0.1950	0.0584

Table 1. Quantitative evaluation results of our validation set in MSE.

To perform quantitative evaluation, we reserve all videos from 9 videos in 2 of the 12 indoor environments from PROX to be a validation set - `BasementSittingBooth`, `N3OpenArea`, and the put the rest into training. We set the input and future sequence length to 5 and 10 frame jump to 6, and an overlapping interval of 11. In total, we have 962 sequences for validation. Though training loss functions differ, validation loss is just a mean over squared distance error of joints for each predicted frame, and should be comparable between models and tasks. We notice that training with shorter sequences and shorter spanning time (5 frames/1 second in, 10 frames/2 seconds out) largely reduces the loss. The RNN slightly outperforms the transformer for both sequence length settings, proving the effectiveness of the modified loss terms. Additionally that both models' prediction losses are less than naive replication of the final frame show that they have learnt something.

5. Conclusion and Future Works

Although we set up the infrastructure to handle the PROX dataset and produce proximity maps for human in the scene, due to time constraints we did not achieve our goal of integrating them as inputs into a predictive model. Instead we train simpler models with just joint locations as input. We had difficulties in RNN and Transformer training, but managed finally to train passable predictive models for 2 second motion prediction.

Future work would incorporate proximity maps and carefully see how much improvement they could add in different scenarios. Comparisons could be made in different types of human motion, with more or less scene context involvement.

For producing proximity maps we use the depth images (as described in 3.2.3) although it would have been possible to artificially create a proximity map with a simulated camera given that we have a full mesh of each scene - this would have the advantage of being able to pick the viewpoint onto the human, as well as being able to overcome occlusion.

To incorporate proximity maps into training, future works could consider applying transfer learning on pretrained vision transformers [10] or video vision transformers [2] to our joint location prediction task. Splitting proximity maps into frames and encoding them as sequences in a spatio-temporal manner could potentially incorporate more temporal scene context to help decode more accurate joint location predictions. Furthermore figuring out how to relate the flat 2D viewpoint of the proximity map to the human body's rotation with respect to the depth camera is important.

A much further away overall end goal is general prediction in video - something similar to the success that large language models have had in predicting future text. Modelling the motion of one human skeleton within an indoor scene context is a very specific example of this more general problem. Does explicit human body modelling have a place in a general framework, along with auxiliary data like human proximity maps? What structure of predictive model could make this work - RNN, Transformer or something else? And what corpus of training data?

Our main takeaways from the project have however been mostly self-educational - insights into RNN and Transformer design, as well as lessons in the complexities of building machine learning systems in this area.

6. Contributions of team members

- Data Pipelining/Preprocessing: Benjamin
- Proximity Map generation: Di, Benjamin
- RNN design/training: Benjamin

- Transformer design/training: Xiyi
- Visualisation Tools: Benjamin, Di, Xiyi

7. Acknowledgements

We thank Siwei Zhang for her supervision on this Virtual Humans (263-5906-00L) course project at ETH Zurich, and Prof. Dr. Siyu Tang, Dr. Sergey Prokudin, Dr. Yan Zhang, Qianli Ma, and Shaofei Wang for their valuable discussion and feedbacks. Part of our training and evaluations are performed on ETH Euler cluster.

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 *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 961–971, 2016. 2

[2] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer, 2021. 6

[3] Zhe Cao, Hang Gao, Karttikeya Mangalam, Qi-Zhi Cai, Minh Vo, and Jitendra Malik. Long-term human motion prediction with scene context, 2020. 2

[4] Hao-Shu Fang, Jinkun Cao, Yu-Wing Tai, and Cewu Lu. Pairwise body-part attention for recognizing human-object interactions, 2018. 1

[5] Katerina Fragkiadaki, Sergey Levine, and Jitendra Malik. Recurrent network models for kinematic tracking. *CoRR*, abs/1508.00271, 2015. 4

[6] A. Gupta, S. Satkin, A. A. Efros, and M. Hebert. From 3d scene geometry to human workspace. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, page 1961–1968, USA, 2011. IEEE Computer Society. 2

[7] Mohamed Hassan, Vasileios Choutas, Dimitrios Tzionas, and Michael J. Black. Resolving 3D human pose ambiguities with 3D scene constraints. In *International Conference on Computer Vision*, pages 2282–2292, Oct. 2019. 1, 2, 4

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 1

[9] Dirk Helbing and Peter Molnar. Social force model for pedestrian dynamics. *Physical Review E*, 51, 05 1998. 2

[10] Alexander Kolesnikov, Alexey Dosovitskiy, Dirk Weissenborn, Georg Heigold, Jakob Uszkoreit, Lucas Beyer, Matthias Minderer, Mostafa Dehghani, Neil Houlsby, Sylvain Gelly, Thomas Unterthiner, and Xiaohua Zhai. An image is worth 16x16 words: Transformers for image recognition at scale. 2021. 6

[11] Donghoon Lee, Sifei Liu, Jinwei Gu, Ming-Yu Liu, Ming-Hsuan Yang, and Jan Kautz. Context-aware synthesis and placement of object instances, 2018. 2

[12] Yong-Lu Li, Xinpeng Liu, Han Lu, Shiyi Wang, Junqi Liu, Jiefeng Li, and Cewu Lu. Detailed 2d-3d joint representation for human-object interaction, 2020. 2

[13] Julieta Martinez, Michael J. Black, and Javier Romero. On human motion prediction using recurrent neural networks. *CoRR*, abs/1705.02445, 2017. 2, 3, 4, 6

[14] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3d hands, face, and body from a single image. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 4, 6

[16] Siwei Zhang, Yan Zhang, Qianli Ma, Michael J. Black, and Siyu Tang. Place: Proximity learning of articulation and contact in 3d environments, 2020. 1, 2