

Spatial Temporal Transformer Networks for Sparse Motion Capture Applications

Lukas Vollenkemper, Martin Kohlhasse

Fachhochschule Bielefeld, Center for Applied Data Science

Schulstraße 10, 33330 Gütersloh

E-Mail: {lukas.vollenkemper, martin.kohlhasse}@fh-bielefeld.de

1 Introduction

Motion Capture technologies provide the possibility to record human motions. In the past, these systems were primarily used by the movie industry, in order to make realistic computer animations. But in recent years their use in the industrial context gained more and more attention. Recording workers motions provides invaluable information that can be used in many applications. These include the improvement of the ergonomics at the workplaces and more sophisticated human-machine interactions [19]. Motion Capture has been done with numerous types of sensor technologies. Optical systems with RGB or Depth Cameras provide an easy to install method [5]. Optical systems can also use infra-red reflectors on the human body as guidance [1]. Inertial systems use glycerometers, accelerometers and electromagnetic trackers to capture human movement [8]. These systems are more suitable to the industrial context since they do not require a line of sight. On the other hand, inertial systems are expensive and need longer setup times due to the large number of sensors needed to provide accurate recordings. For this reason researchers have developed different solutions to make motion capture recordings from a small number of sensors. This is called sparse motion capture. By reducing the number of sensors needed for accurate measurement, two of the major challenges regarding inertial systems are tackled. It reduces the costs of the system as well as the setup time required in preparation of a recording. Thus, sparse motion capture

DOI: 10.58895/ksp/1000151141-5 erschienen in:

Proceedings – 32. Workshop Computational Intelligence: Berlin, 1. - 2. Dezember 2022

DOI: 10.58895/ksp/1000151141 | <https://www.ksp.kit.edu/site/books/m/10.58895/ksp/1000151141/>

is a possibility to foster the usage of motion capture systems in the industrial context. In the past recurrent neural networks as well as transformer networks have been used for sparse motion capture setups. Transformer networks use attention layers to model relationships between input elements. In this work the performance of an alternative attention formulation, known as spatial temporal attention inside of the transformer networks is tested.

2 Sparse Motion Capture

There are approaches to sparse motion capture from different research areas. Some rely on kinematic body models [3]. But many others use machine learning approaches to accomplish sparse motion capture. In [5], an approach to sparse motion capture based on neural networks is presented. A recurrent neural network (RNN) is trained to estimate a pose from only six sensors. Instead of directly determining the position and orientation of individual limbs, parameters of a simplified human body model, the Skinned Multi-Person Linear Model [12] (SMPL) are determined. The authors show that real-time prediction is possible using the approach. It is also shown that a bi-directional RNN architecture produces better results than a classical RNN.

Long-Short-Term Memory Networks (LSTM) as a transformer network are used in [4]. The authors furthermore provide a public dataset with motion capture recordings. Data from six sensors serve as input in this approach as well. Here, the sensors sit on the hips, forearms, head, and knees. In the experiments, the LSTM and Transformer architectures are tested against each other with different sensor configurations. Both architectures show good results. The best configuration in their test is sternum, both wrists, lower legs and hip.

In [2] a RNN is trained to generate input parameters for the SMPL body model, similar to [5]. Their approach not only predicts the human pose, but also provides an uncertainty estimation. Their sensor setup consists of six sensors at the wrists, lower legs, head and hip. Another approach to sparse motion capture is presented in [11]. Here, an ensemble of LSTM models is used. Six sensors are used as well, but here the sensors are located at the forearms and at the feet instead of wrists and lower legs. The ensemble consists of bi-directional LSTM

networks, each of which provides an estimate of the pose based on information from one sensor. The individual estimates are then combined to estimate the pose.

In [14], graph convolutional neural networks (GCNN) are tested as an alternative to the RNNs used in [5]. The sparse setup of the sensors is identical and parameters of the SMPL body model are also used as the target. The GCNN allows the topology of the human skeleton to be directly represented in the model. Thus, the authors show that they achieve more accurate pose estimation results than the reference architecture of [5].

In [7], a method is presented that also uses the SMPL model and determines the input parameters based on the electromagnetic signals. They use a recurrent neural network to predict the SMPL Parameters, but the method is limited to electromagnetic sensors.

3 Transformer Networks

Transformer networks and the attention mechanism have become popular since [16] showed that similar or better performance was achieved over recurrent networks in the area of language processing. At the same time, transformer architectures offer the advantage of being parallelizable, which significantly reduces training times. This makes transformer networks an interesting alternative to the recurrent architectures discussed above. The most important building block of the Transformer architecture is the attention mechanism. Via attention, the network learns to understand relationships between individual parts of the sequence. In the language processing domain, this could be the relationship between subject and predicate or verb and adverb. The most widely used version of the attention mechanism is the scaled-dot-product attention described in [16]). The way it works is shown in equations (1-6) The input for each attention layer is a matrix $\mathbf{X} \in \mathbb{R}^{S \times d}$, where S denotes the sequence length and d the length of each sequence element. The layer has three learned parameters $\mathbf{W}_{\text{value}} \in \mathbb{R}^{d \times d_{\text{value}}}$, $\mathbf{W}_{\text{key}} \in \mathbb{R}^{d \times d_{\text{key}}}$ and $\mathbf{W}_{\text{query}} \in \mathbb{R}^{d \times d_{\text{key}}}$. The query, key and value matrices can then be calculated as

$$\mathbf{V} = \mathbf{X}\mathbf{W}_{\text{value}}, \quad \mathbf{K} = \mathbf{X}\mathbf{W}_{\text{key}}, \quad \mathbf{Q} = \mathbf{X}\mathbf{W}_{\text{query}} \quad (1)$$

In order to calculate the attention matrix the key and query matrices are multiplied and scaled with the square root of d_{key} . The scaling is done mainly for numeric stability [16]. This yields the logits matrix \mathbf{L} as described in equation (2). In the later application it is sometimes necessary to prevent certain information to flow through the attention layer. Thus the attention can be masked with a mask $\mathbf{M} \in \mathbb{R}^{S \times S}$ (equation (3)). That is a simple matrix with ones in the lower triangle and negative infinity values in the upper triangle. It is important to note that this is an optional operation that can be skipped.

$$\mathbf{L} = \frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_{\text{key}}}} \quad (2)$$

$$\mathbf{L}_{\text{masked}} = \mathbf{L} \odot \mathbf{M} \quad (\text{optional}) \quad (3)$$

Afterwards a softmax operation is done over each row of the logits matrix (4, 5). The output of this softmax operation is called attention weights. They are multiplied with \mathbf{V} to get the encodings of the attention layer.

$$\sigma_i(z) := \frac{e^{z_i}}{\sum_{j=1}^S e^{z_j}} \quad (4)$$

$$\mathbf{S} = (\sigma(\mathbf{l}_1), \sigma(\mathbf{l}_2), \dots, \sigma(\mathbf{l}_S))^T \quad (5)$$

$$\mathbf{O} = \mathbf{S}\mathbf{V}\mathbf{W}_{\text{Out}} \quad (6)$$

The output of the softmax \mathbf{S} multiplied by \mathbf{V} will be of shape $\mathbb{R}^{S \times d_{\text{value}}}$. Another linear projection $\mathbf{W}_{\text{Out}} \in \mathbb{R}^{d_{\text{value}} \times d}$ is used to receive the final output of the attention layer, which will have the same size as the input sequence \mathbf{X} (equation (6)). In [16], it is proposed to train multiple parallel attention heads rather than performing this process only once. Each head thereby has its own projections for \mathbf{Q} , \mathbf{K} , and \mathbf{V} and can thus focus on detecting specific relation types. This is usually called multi-head attention. Standard transformer networks follow an encoder-decoder structure shown in Figure 1.

On the left is the encoder block, into which the input sequences are given. The right block is the decoder, which processes the encoder results together with the previous output variables. In Positional Encoding, information about

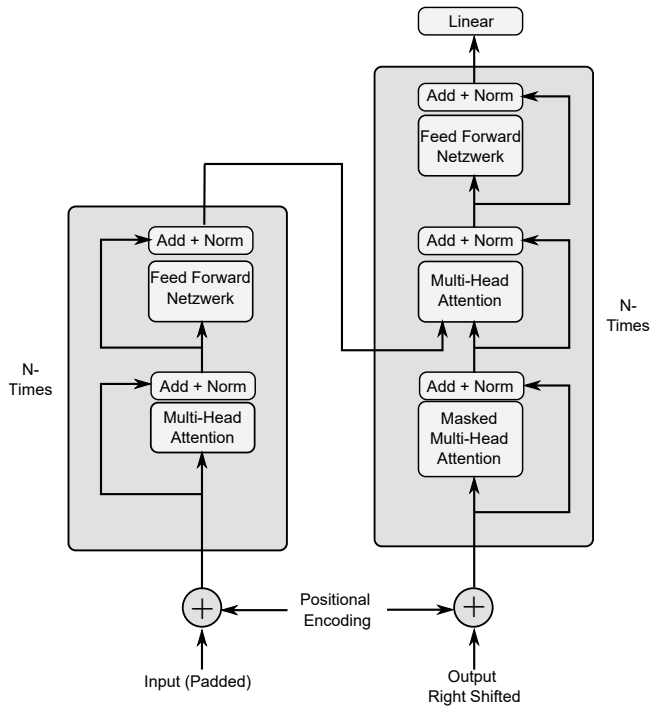


Figure 1: Architecture of the transformer network, [16]

the position of each element in the sequence is added to the network. This is necessary because transformers, unlike recurrent networks, perform the individual prediction steps in parallel during training. The inherent order of the input data is therefore lost. Usually, sine and cosine functions are used as positional encoding [16]. In the encoder, the input sequences are processed by an attention layer and a feed-forward network. This process can be repeated several times before the processed input sequence is passed to the decoder. In the decoder, the previous outputs are processed first. Since the entire sequence is processed in parallel, the values are masked in the attention layer. By masking, only information from previous time steps is used for each time step. This is followed by another attention layer where the encoder results are used as attention. More precisely, the encoder results are used as a inputs

to compute the \mathbf{K} and \mathbf{V} matrices, while the results of the previous masked decoder attention are used to compute \mathbf{Q} . Then, as in the encoder, a feed-forward network is also used. The decoder blocks are also stackable. Skip connections are present throughout the network, where the input of a layer is added to its output. The result is then batch normalized [6]. During training, the encoder block is given the input sequences. The decoder block also receives the sequences, but shifted one position to the right. In the first decoder attention layers, the attention weights are masked such that only previous values can be used to determine the respective output. Through this masking, the training can be parallelized, which significantly increases the training speed. The encoder creates an encoded version of the input sequence. The decoder then creates the prediction of the next element in the sequence based on this encoding and the previous elements.

4 Spatial Temporal Transformer Networks

In the original transformer architecture it is assumed that each element of the in- and output sequences can be described with a single vector. In NLP these vectors usually come from word embedding, where the single dimensions are not easily interpretable. However in motion capture applications the sequence elements have an inherent structure. They can be grouped by point on the human body, where each point can have a number of measurement values (coordinates, acceleration values, orientation quaternions, etc.). This structure is ignored when the input and output measurements are simply flattened to a vector. *Spatial-Temporal Transformers* (STT) therefore extend the classical transformer architecture by splitting the encoder block into two separate streams that independently consider spatial and temporal aspects of the input data. In the spatial stream, the parameters for each joint are processed in relation to all other joints at the same time. In the temporal stream, the parameters of each joint are analyzed over time, independently of the other joints. STT networks are already used for various tasks. In [9], an STT network for image processing is presented. Other architectures deal with the use in traffic modeling [18] or crime analysis [17]. STT networks are also already being used in human motion analysis, particularly in the area of action recognition.

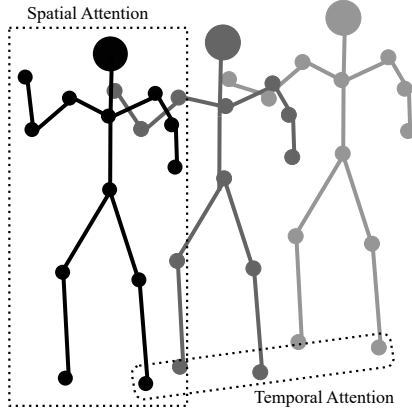


Figure 2: Illustration of the split in temporal and spatial properties

In [20], an approach to action recognition is presented that is particularly robust to noise or missing information. Here, additional tokens are given to each joint at each time step, such as what type of joint it is and the number of the time step.

In [13] Graph Convolutional Networks (GCN) and Convolutional Networks (CNN) from [15] are used to infer such information from the data directly. Furthermore, the authors present the spatial self-attention (spatial encoder) and temporal self attention (temporal encoder) layers. These layers will be tested as an alternative to the classic attention mechanism for sparse motion capture.

Inputs to the spatial encoder are the joint parameters $\mathbf{X} \in \mathbb{R}^{J \times T \times C_{in}}$. Here T denotes the time steps, J the joints, and C_{in} the number of measurements per joint. The temporal encoder takes $\mathbf{X}^T \in \mathbb{R}^{T \times J \times C_{in}}$ as inputs. The same formulas as in (1-6) apply to both encoders. However the size of the individual matrices changes slightly. When multiplying three-dimensional tensors with matrices the n-Mode product with $n = 1$ is used [10]. \mathbf{W}_{key} and \mathbf{W}_{query} are now in $\mathbb{R}^{C_{in} \times d_{key}}$ and \mathbf{W}_{value} in $\mathbb{R}^{C_{in} \times d_{value}}$. The output projection is a matrix \mathbf{W}_{out} in $\mathbb{R}^{d_{value} \times C_{in}}$ in both cases. For the spatial encoder the result of formula (2) will be in $\mathbb{R}^{T \times J \times J}$. That way the network models the importance of different body

joints for each other per time frame. For the temporal encoder however the logits will be in $\mathbb{R}^{J \times T \times T}$. The model can use this to select important time steps for each single time step.

While this split introduces spatial and temporal versions of the projection matrices, the matrices will often be smaller. The number of learned parameters in the classical attention module is $J C_{\text{in}} (2d_{\text{value}} + 2d_{\text{key}})$. That is because the C_{in} measurements per joint J have to be described in a single vector in the former version version. Number of learned parameters in the spatio-temporal formulation is $2C_{\text{in}} (2d_{\text{key}} + 2d_{\text{value}})$. Thus, the spatio-temporal version uses less parameters as long as more than two joints are analyzed and the key and query dimension is the same. Another useful property is that the number of parameters does not change with the number of joints, as in the spatio-temporal attention formulation. But it also introduces a limitation of this formulation: the same matrices are used over all joints or time steps which might be a limitation to the accuracy.

5 STT for Sparse Motion Capture

After STT networks have been successfully applied in the field of action recognition, it is promising to apply them to pose estimation as well, since they already showed to be able to analyze human motion well. Especially when generating realistic sequences by sparse motion capture, the captured pose must also make sense in combination with the other poses of the sequence. Explicitly modeling the motion of individual joints over time should improve results here. By modeling the spatial relationships in each time segment, the network gets the opportunity to learn and map the structure and anatomy of the human body. To test the applicability of the attention modules presented in [13] they are integrated into the classical Transformer architecture. This is illustrated in Figure 4.

The spatial- and temporal attention layers replace here the classical self attention. Because in this case two layers are used, whose output size corresponds again to the input size, the results must be merged for the skip connection. In this work the outputs of both attention layers are added. In the decoder,

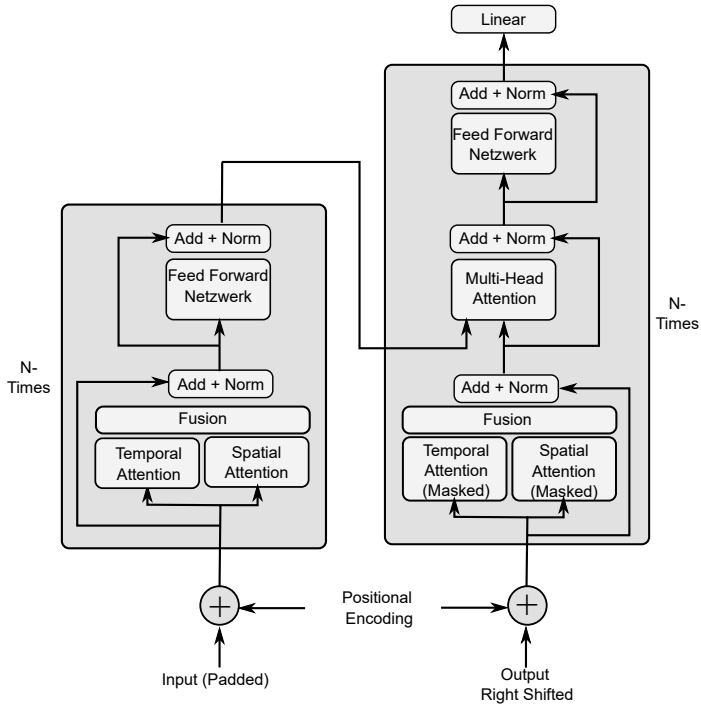


Figure 3: Integration of the spatio temporal encoders into the transformer network structure

temporal attention must be masked so that only the previous values are used for each time point. In the spatial stream the masking is done over the time dimension, such that only the encoding from previous times steps are accessible for the prediction of the current time step. Subsequently, the classical attention mechanism is used to integrate the information from the encoder in the decoder. Since the results from spatial and temporal attention are already combined in the decoder, no further subdivision is done. The rest of the architecture is identical to the classical transformer.

6 Data Sources

Two data sources are used in the experiments. First, Virginia Tech University (VTU) provides a dataset described in [4]. This contains data from 17 participants who were observed using an XSense motion capture suit. Four participants were employees at a hardware store who recorded part of their shift (specifically material handling in the warehouse). The 13 remaining participants were VTU students who recorded their daily routine on campus. A wide variety of activities have been performed. From sitting in the seminar, to walking and standing, to doing sports exercises. This results in a total of approximately 40 hours of motion capture recordings with a resolution of 240 measurements per second.

Secondly, recordings were made with the same system at CfADS Gütersloh. One subject was observed for a total of 3 hours and 47 minutes. This also included a variety of tasks to ensure the greatest possible variance. Among others, these included sports exercises, household activities, working at a desk and eating.

All data has been read from the X-Sense system and reduced to 8 measurements per second. The X-Sense data then contains the position of a total of 64 points on the body. In this work, 27 of these points are used as output variables. These points were selected because they are needed for a later application. The X-Sense system provides position values as well as acceleration or rotation measurements. The approach presented here uses only the 3d coordinates of all 27 points. Thus, it can be easily transferred to other motion capture systems, e.g. optical. In total, 225466 training sequences with 32 elements each are created. In addition, there are 6704 validation sequences that are not used in the training. Six sensor positions with three coordinates each serve as input variables. The output variables are the 27 positions with the corresponding coordinates. Note that this means that the input values are part of the target positions as well. This setup is similar to [4] and could be useful in further research to model noisy input data.

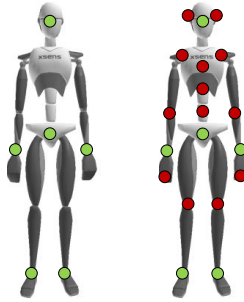


Figure 4: Input of the network (left) and desired outputs (right)

7 Experiments and Benchmark

In this section, the presented architecture will be compared with a classical transformer with respect to performance in sparse motion capture. In [4] it is already shown that the classical transformer architecture can achieve good results in sparse motion capture tasks. The code was used in order to generate predictions from a classical transformer architecture, denoted as VT NMP (Virginia Tech Natural Motion Processing). For this purpose, the two models are first trained five times with different (random) initialization of the parameters. The results shown here are based on the model with the mean validation error. Both models are initialized with three attention heads and the feed forward networks have 128 neurons in the hidden layers. Dropout of 0.01 is used in the attention layers to make the training more robust. The input data is the position of the hips, ankles, wrists and head.

In order to test the model performance a inference procedure slightly different from the forward pass in the training is used. As mentioned above during the training the transformer receives a sequence of target values. In the later application however these values will not be known. Instead the models always predict the next element of the sequence, with their former predictions as target vector. For the first element a padding vector is used as target input. The test is done with all sequences in the validation data set with both a vtntp model and the spatio-temporal transformer. The mean quadratic error of the estimated

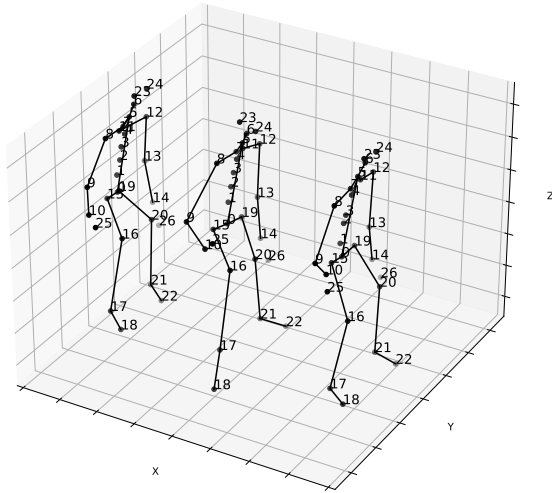


Figure 5: From left to right: VTNMP Prediction, Actual Pose, STT Prediction

point positions per sequence is shown in Figure 6. The error reported can not be interpreted in centimeter, since the batch normalization layers in the networks change the scale. The STT shows a larger error on average as well as a larger number of outliers with large errors.

Figure 7 compares the mean squared error of both networks regarding the single joints over all validation sequences. Points on arms and legs are more difficult to estimate than the spine and hip positions. Hands and toes are the most challenging body parts. STT has problems predicting the position of the right hand. A possible explanation is that the right hand is used in a larger variety of situations. STT also produces large errors at positions, that were originally given to the network (Feet and Hands). This problem in copying of the information of certain joints might be due to the fact that STT uses the same matrices for all joints, which differs from the classical transformer, where each joint, measurement combination has its own parameter in the query, key and value projections.

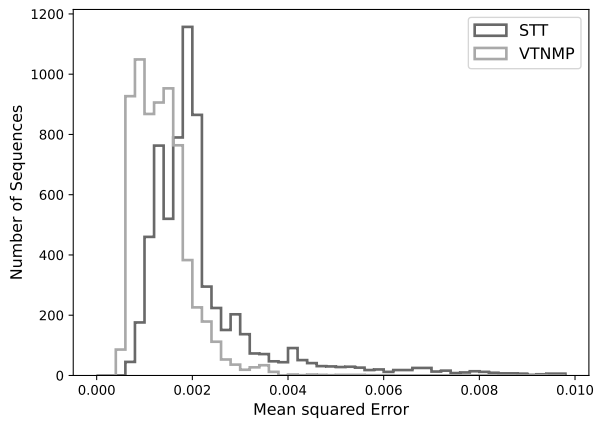


Figure 6: Mean Squared error per Test-Sequence

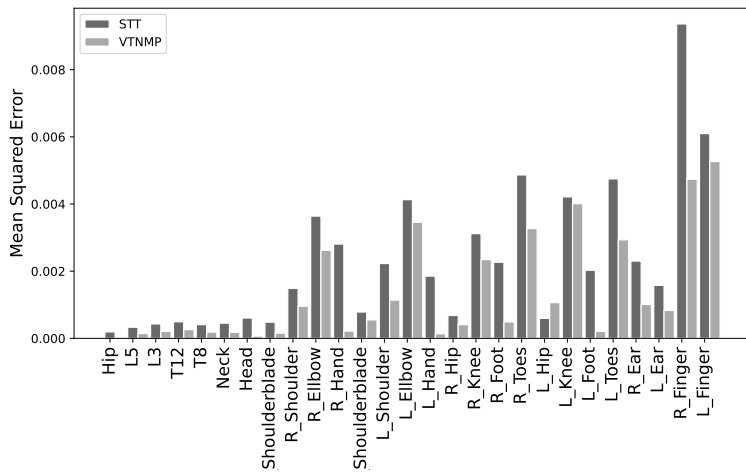


Figure 7: Error per Body Part

8 Conclusion

The aim of this article was to test the applicability of spatio-temporal transformer architectures in sparse pose estimation problems. STT architectures had shown good results in other areas of human motion analysis, especially action recognition. However, it must be concluded that they are not a useful replacement for the multi-head attention layers for sparse motion capture. The performance was tested against an approach using the vanilla attention layers. STT showed that it produces a higher placement error on average. Therefore it must be concluded that analyzing the spatial and temporal information separately might prevent the network from depicting some characteristics of human motion, that are crucial to predict human motion. A possible research direction are more sophisticated versions of the fusion of both streams to enhance the information exchange between temporal and spatial layers. If this hurdle is overcome, the spatial-temporal attention layers provide a useful instrument, because their parameter space does not grow with the number of joints.

References

- [1] Accuracy of human motion capture systems for sport applications; state-of-the-art review. *European journal of sport science*, 18(6):806–819, 2018.
- [2] Hammad Tanveer Butt, Bertram Taetz, Mathias Musahl, Maria A. Sanchez, Pramod Murthy, and Didier Stricker. Magnetometer robust deep human pose regression with uncertainty prediction using sparse body worn magnetic inertial measurement units. *IEEE Access*, 9:36657–36673, 2021.
- [3] Karsten Eckhoff, Manon Kok, Sergio Lucia, and Thomas Seel. Sparse magnetometer-free inertial motion tracking – a condition for observability in double hinge joint systems. *IFAC-PapersOnLine*, 53(2):16023–16030, 2020.

- [4] Jack H. Geissinger and Alan T. Asbeck. Motion inference using sparse inertial sensors, self-supervised learning, and a new dataset of unscripted human motion. *Sensors (Basel, Switzerland)*, 20(21), 2020.
- [5] Yinghao Huang, Manuel Kaufmann, Emre Aksan, Michael J. Black, Otmar Hilliges, and Gerard Pons-Moll. Deep inertial poser. *ACM Transactions on Graphics*, 37(6):1–15, 2018.
- [6] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.
- [7] Manuel Kaufmann, Yi Zhao, Chengcheng Tang, Lingling Tao, Christopher Twigg, Jie Song, Robert Wang, and Otmar Hilliges. Em-pose: 3d human pose estimation from sparse electromagnetic trackers. pages 11510–11520, 2021.
- [8] Sunwook Kim and Maury A. Nussbaum. Performance evaluation of a wearable inertial motion capture system for capturing physical exposures during manual material handling tasks. *Ergonomics*, 56(2):314–326, 2013. PMID: 23231730.
- [9] Woojae Kim, Jongyoo Kim, Sewoong Ahn, Jinwoo Kim, and Sanghoon Lee. Deep video quality assessor: From spatio-temporal visual sensitivity to a convolutional neural aggregation network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.
- [10] Kolda, T. and Bader B. Tensor Decompositions and Applications In *SIAM Review* Vol. 51 n. 3, pages 455-500, 2009.
- [11] Deepak Nagaraj, Erik Schake, Patrick Leiner, and Dirk Werth. An rnn-ensemble approach for real time human pose estimation from sparse imus. In Nicolai Petkov, editor, *Proceedings of the 3rd International Conference on Applications of Intelligent Systems*, ACM Digital Library, pages 1–6, New York, NY, United States, 2020. Association for Computing Machinery.
- [12] Georgios Pavlakos, Luyang Zhu, Xiaowei Zhou, and Kostas Daniilidis. Learning to estimate 3d human pose and shape from a single color image.

- [13] Chiara Plizzari, Marco Cannici, and Matteo Matteucci. Spatial temporal transformer network for skeleton-based action recognition. In *Pattern Recognition. ICPR International Workshops and Challenges*, pages 694–701. Springer International Publishing, 2021.
- [14] Patrik Puchert and Timo Ropinski. Human pose estimation from sparse inertial measurements through recurrent graph convolution.
- [15] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. *undefined*, 2019.
- [16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz 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.
- [17] Xian Wu, Chao Huang, Chuxu Zhang, and Nitesh V. Chawla. Hierarchically structured transformer networks for fine-grained spatial event forecasting. In Yennun Huang, Irwin King, Tie-Yan Liu, and Maarten van Steen, editors, *WWW '20*, pages 2320–2330, New York, op. 2020. ACM =Association for Computing Machinery.
- [18] Mingxing Xu, Wenrui Dai, Chunmiao Liu, Xing Gao, Weiyao Lin, Guo-Jun Qi, and Hongkai Xiong. Spatial-temporal transformer networks for traffic flow forecasting.
- [19] Zuhair Zafar, Rahul Venugopal, and Karsten Berns. Real-time recognition of human postures for human-robot interaction. 2018.
- [20] Yuhan Zhang, Bo Wu, Wen Li, Lixin Duan, and Chuang Gan. Stst: Spatial-temporal specialized transformer for skeleton-based action recognition. In Heng Tao Shen, editor, *Proceedings of the 29th ACM International Conference on Multimedia*, ACM Digital Library, pages 3229–3237, New York,NY,United States, 2021. Association for Computing Machinery.